

Editor:
Bipin C. Desai

General Chairs:
Bipin C. Desai
Jefferey Ullman

Program Chairs:
Richard McClatchey
Motomichi Toyoma



IDEAS 2021

25th

International Database Applications & Engineering Symposium

Concordia University, Montreal, Canada

2021-07-14 – 2021-07-16

(On-line - Video conference)

Proceedings Editor

Bipin C. Desai, Concordia University, Canada

General Chairs

Bipin C. Desai, Concordia University, Canada

Jeffrey Ullman, Stanford University, USA

Program Chairs

Richard McClatchey, Univ. Of West England, U.K

Motomichi Toyoma, Keio University, Japan

The Association for Computing Machinery

2 Penn Plaza, Suite 701
New York, New York 10121-0701

ACM COPYRIGHT NOTICE. Copyright © 2008 by the Association for Computing Machinery, Inc. Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honoured. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Publications Dept., ACM, Inc., fax +1 (212) 869-0481, or permissions@acm.org

ACM intends to create a complete electronic archive of all articles and/or other material previously published by ACM. If you have written a work that was previously published by ACM in any journal or conference proceedings prior to 1978, or any SIG Newsletter at any time, and you do NOT want this work to appear in the ACM Digital Library, please inform permissions@acm.org, stating the title of the work, the author(s), and where and when published.

ISBN: For printed proceedings 978-1-988392-07-3

Editorial production by: BytePress

Cover Artwork by: Logo Copyright Lettie

Table of Content

Invited Papers
Full Papers
Short Papers
Foreword
Preface
Reviewers from the Program Committee
External Reviewers
Organizers

Invited Paper

SQL-like query language and referential constraints on tree-structured data	1
Foto N Afrati(National Technical University of Athens)	
Matthew George Damigos(National Technical University of Athens)	
Nikos Stasinopoulos(National Technical University of Athens)	
 A Framework for Enhancing Deep Learning Based Recommender Systems with Knowledge Graphs	 11
Sudhir P. Mudur(Concordia University)	
Serguei A Mokhov(Concordia University)	
Yuhao Mao(Concordia University)	
 Sink Group Between Centrality	 21
Evangelia Fragkou(University of Thessaly)	
Dimitrios Katsaros(University of Thessaly)	
Yannis Manolopoulos(Open University of Cyprus)	

Full Paper

Bringing Common Subexpression Problem from the Dark to Light: Towards Large-Scale Workload Optimizations	27
Mohamed Kechar(Universite d'Oran Es-Senia)	
Ladjel Bellatreche(Ecole Nationale Supérieure de Mécanique et d'Aéronautique)	
Safia Nait-bahloul(Universite d'Oran Es-Senia)	
Colonization of the internet	36
Bipin C. Desai(Concordia University)	
Data Mining Autosomal Archaeogenetic Data to Determine Ancient Migrations in the Aegean Area	47
Peter Z. Revesz(University of Nebraska - Lincoln)	
MANTIS: Multiple Type and Attribute Index Selection using Deep Reinforcement Learning	57
Vishal Sharma(Utah State University)	
Curtis Dyreson(Utah State University)	
Nicholas Flann(Utah State University)	
Explainable data analytics for disease and healthcare informatics	66
Carson K. Leung(University of Manitoba)	
Daryl L.x. Fung(University of Manitoba)	
Daniel Mai(University of Manitoba)	
Joglas Souza(University of Manitoba)	
Jason Tran(University of Manitoba)	
Qi Wen(University of Manitoba)	
ICIX: a Semantic Information Extraction System	76
Angel Luis Garrido(Universidad de Zaragoza)	
Alvaro Peiro(Request Pending)	
Cristian Roman(InSynergy Consulting S.A.)	
Carlos Bobed(Universidad de Zaragoza)	
Eduardo Mena(Universidad de Zaragoza)	
Software Quality Assessment of a Web Application for Biomedical Data Analysis	85

Full Paper(Continued)

Kristina Lietz(Fraunhofer ITEM)

Ingmar Wiese(Fraunhofer ITEM)

Lena Wiese(Fraunhofer ITEM)

A Zone-Based Data Lake Architecture for IoT, Small and Big Data

95

Yan Zhao(Universite Toulouse (Capitole))

Imen Megdiche(Institut de Recherche en Informatique de Toulouse)

Franck Ravat(Universite Toulouse (Capitole))

Vincent-nam Dang(Institut de Recherche en Informatique de Toulouse)

COVID-19 Concerns in US: Topic Detection in Twitter

104

Carmela Comito(Consiglio Nazionale delle Ricerche)

IDEAS: the first quarter century

112

Bipin C. Desai(Concordia University)

Load Balanced Semantic aware RDF Graph

127

Ami Pandat(Dhirubhai Ambani Institute for Information and Communication Technology)

Nidhi Gupta(Dhirubhai Ambani Institute for Information and Communication Technology)

Minal Bhise(Dhirubhai Ambani Institute for Information and Communication Technology)

Categorical Management of Multi-Model Data

134

Martin Svoboda(Charles University Prague)

Pavel Contos(Charles University Prague)

Irena Holubova(Charles University Prague)

Exploratory analysis of methods for automated classification of clinical diagnoses in Veterinary Medicine

141

Oscar Tamburis(University of Naples Federico II)

Elio Masciari(Consiglio Nazionale delle Ricerche)

Gerardo Fatone(University of Naples Federico II)

Optimizing Transaction Schedules on Universal Quantum Computers via Code Generation for Grover's Search Algorithm

149

Sven Groppe(Medizinische Universitat Lubeck)

Full Paper(Continued)

Jinghua Groppe(Medizinische Universitat Lubeck)

Customized Eager-Lazy Data Cleansing for Satisfactory Big Data Veracity

157

Soror Sahri(Universite Rene Descartes (Paris V))

Rim Moussa(Universite de 7 Novembre a Carthage)

Measuring Quality of Workers by Goodness-of-Fit of Machine Learning Model in Crowdsourcing

166

Yu Suzuki(Gifu University)

SSstory: 3D data storytelling based on SuperSQL and Unity

173

Jingrui Li(Keio University)

Motomichi Toyama(Keio University)

Kento Goto(Keio University)

Proposing a Business View of Enterprise Data

184

Max Chevalier(Institut de Recherche en Informatique de Toulouse)

Joan Marty

Franck Ravat(Institut de Recherche en Informatique de Toulouse)

Bastien Vidé(Institut de Recherche en Informatique de Toulouse)

Analysis-oriented Metadata for Data Lakes

194

Yan Zhao(Universite Toulouse (Capitole))

Julien Aligon(Universite des Sciences Sociales (Toulouse I))

Franck Ravat(Universite Toulouse (Capitole))

Chantal Soule-dupuy(Universite Toulouse (Capitole))

Gabriel Ferrettini(Universite Toulouse (Capitole))

Imen Megdiche(Institut de Recherche en Informatique de Toulouse)

Priority-Based Skyline Query Processing for Incomplete Data

204

Chuan-Ming Liu(National Taipei University of Technology)

Denis Pak

Ari Ernesto Ortiz Castellanos(National Taipei University of Technology)

Looking for Jobs? Matching Adults with Autism with Potential Employers for Job Opportunities

212

Full Paper(Continued)

Joseph Bills(Brigham Young University)

Yiu-kai Dennis Ng(Brigham Young University)

On Efficiently Equi-Joining Graphs

222

Giacomo Bergami(Free University of Bozen)

An Automatic Schema-Instance Approach for Merging Multidimensional Data Warehouses

232

Yuzhao Yang(Universite Toulouse (Capitole))

Jérôme Darmont(Université de Lyon)

Franck Ravat(Universite Toulouse (Capitole))

Olivier Teste(Universite de Toulouse-le-Mirail (Toulouse II))

Multi-Model Data Modeling and Representation: State of the Art and Research Challenges

242

Pavel Contos(Charles University Prague)

Irena Holubova(Charles University Prague)

Martin Svoboda(Charles University Prague)

ArchaeoDAL: A Data Lake for ArchaeologicalData Management and Analytics

252

Pengfei Liu(Université de Lyon)

Sabine Loudcher(Universite Lyon 2)

Jérôme Darmont(Université de Lyon)

Camille Noûs

Towards a continuous forecasting mechanism of parking occupancy in urban environments

263

Miratul Khusna Mufida(Univ.Poly. Hauts-de-France)

Abdessamad Ait El Cadi(Univ.Poly. Hauts-de-France)

Thierry Delot(Univ.Poly. Hauts-de-France)

Martin Trépanier(Polytechnic School of Montreal)

Short Paper

Predicting late symptoms of head and neck cancer treatment using LSTM and patient reported outcomes	273
Yaohua Wang(University of Iowa)	
Guadalupe M Canahuate(University of Iowa)	
Lisanne V Van Dijk(University of Texas M.D. Anderson Cancer Center)	
Abdallah S. R. Mohamed(University of Texas M.D. Anderson Cancer Center)	
Clifton David Fuller(University of Texas M.D. Anderson Cancer Center)	
Xinhua Zhang(University of Illinois at Chicago)	
Georgeta-Elisabeta Marai(University of Illinois at Chicago)	
 Data Management in the Data Lake: A Systematic Mapping	 279
Firas Zouari(Universite Jean Moulin (Lyon III))	
Nadia Kabachi(Universite Claude Bernard (Lyon I))	
Khouloud Boukadi(Universite de Sfax)	
Chirine Ghedira Guegan(Universite Jean Moulin (Lyon III))	
 Rigorous Measurement Model for Validity of Big Data: MEGA Approach	 284
Dave Bhardwaj(Concordia University)	
Juan J. Cuadrado-gallego And Olga Ormandjieva(Concordia University)	
 Viral pneumonia images classification by Multiple Instance Learning: preliminary results	 291
Ester Zumpano(University of Calabria)	
Antonio Fuduli(University of Calabria)	
Eugenio Vocaturo(University of Calabria)	
Matteo Avolio(University of Calabria)	
 An In-Browser Collaborative System for Functional Annotations	 296
Yui Saeki(Keio University)	
Motomichi Toyama(Keio University)	
 Sentimental Analysis Applications and Approaches during COVID-19: A Survey	 303
Areeba Umair(University of Naples Federico II)	
Elio Masciari(Consiglio Nazionale delle Ricerche)	
Muhammad Habib Ullah(University of Naples Federico II)	

Preface

These are the proceedings of the 25th annual event of IDEAS: it marks a quarter century of organizing and holding these meetings in the series. This meeting had the further challenge of dealing, for the second year, with the Corona virus pandemic. Having endured the global lock down in 2020, we were looking forward to get together for this silver anniversary event in cosmopolitan Montreal. Unfortunately the second and the third waves of covid called for a change in the mode of the meeting late in the call for papers giving us a shorter time for the final sprint! It is ironic that IDEAS in its span of 25 years had to contend first with SARS and, two years in a row with Covid. SARS was contained and we were able to hold an in-person meeting in Hong Kong; however, this time we had to switch, second year in a row, during the last weeks of the CFP due to the unending waves of the pandemic. It is heartening to learn that in spite of these challenges, we received 63 submissions plus 3 invited papers. This allowed us to continue to be selective! This meeting highlights the current pre-occupation with AI, big data, block chain, data analytics, machine learning, the issues of a pandemic and the tyranny of the web; this is reflected in the accepted papers in these proceedings.

We would like to take this opportunity to thank the members of our program committee, listed here, for their help in the review process. All the submitted papers were assigned to four reviewers and we got back over 3.2 reviews on the average due to the shorter review periods. The proceedings consist of 27 full papers(acceptance rate 43%), and 6 short papers (9%) .

Acknowledgment: This conference would not have been possible without the help and effort of many people and organizations. Thanks are owed to:

- ACM (Anna Lacson, Craig Rodkin, and Barbara Ryan),
- BytePress, ConfSys.org, Concordia University (Will Knight and Gerry Laval),
- Many other people and support staff, who contributed selflessly have been involved in organizing and holding this event.

We appreciate their efforts and dedications.

Bipin C. Desai
Concordia University
Canada

Richard McClatchey
Univ. Of West England
U.K

Motomichi Toyoma,
Keio University
Japan

Jeffrey Ullman
Stanford University
USA

Montreal, July, 2021

Reviewers from the Program Committee

Foto Afrati (National Technical University of Athens, Greece)
Toshiyuki Amagasa (Tsukuba University, Japan)
Masayoshi Aritsugi (Kumamoto University, Japan)
Gilbert Babin (HEC Montreal, Canada)
Giacomo Bergami (Free University of Bozen, Italy)
Jorge Bernardino (Instituto Politecnico de Coimbra, Portugal)
Christophe Bobineau (Institut National Polytechnique de Grenoble, France)
Francesco Buccafurri (University of Reggio Calabria, Italy)
Gregory Butler (Concordia University, Canada)
Luciano Caroprese (University of Calabria, Italy)
Richard Chbeir (Universite de Pau et des Pays de l'Adour, France)
Rui Chen (Samsung, United States)
David Chiu (University of Puget Sound, United States)
Martine Collard (Universite des Antilles, France)
Carmela Comito (Consiglio Nazionale delle Ricerche, Italy)
Gabriel David (Universidade do Porto, Portugal)
Marcos Domingues (Universidade Estadual de Maringa, Brazil)
Brett Drury (Universidade do Porto, Portugal)
Magdalini Eirinaki (San Jose State University, United States)
Markus Endres (Universitat Augsburg, Germany)
Bettina Fazzinga (Consiglio Nazionale delle Ricerche, Italy)
Sergio Flesca (University of Calabria, Italy)
Alberto Freitas (Universidade do Porto, Portugal)
Filippo Furfaro (University of Calabria, Italy)
Sven Groppe (Medizinische Universitat Lubeck, Germany)
Marc Gyssens (Hasselt University, Belgium)
Irena Holubova (Charles University Prague, Czech Republic)
Mirjana Ivanovic (University of Novi Sad, Serbia and Montenegro)
Sotirios Kontogiannis (University of Ioannina, Greece)

Gerry Laval (ConfSys.org, Canada)
Carson Leung (University of Manitoba, Canada)
Chuan-ming Liu (National Taipei University of Technology, Taiwan)
Grigorios Loukides (King's College London, University of London, United Kingdom)
Bertil Marques (Instituto Superior de Engenharia do Porto, Portugal)
Giuseppe Mazzeo (Facebook, United States)
Richard McClatchey (University of the West of England, Bristol, United Kingdom)
Noman Mohammed (University of Manitoba, Canada)
Yang-sae Moon (Kangwon National University, Korea Republic)
Kamran Munir (University of the West of England, Bristol, United Kingdom)
Yiu-kai Ng (Brigham Young University, United States)
Wilfred Ng (Hong Kong University of Science and Technology, Hong Kong)
Paulo Oliveira (Instituto Superior de Engenharia do Porto, Portugal)
Olga Ormandjieva (Concordia University, Canada)
Valéria Pequeno (Universidade Autônoma de Lisboa Luís de Camões, Portugal)
Giuseppe Polese (University of Salerno, Italy)
Luboš Popelínský (Masaryk University, Czech Republic)
Filipe Portela (Universidade do Minho, Portugal)
Venkatesh Raghavan (Pivotal Corporation, United States)
Peter Revesz (University of Nebraska - Lincoln, United States)
Marina Ribaud (University of Genoa, Italy)
Antonino Rullo (University of Calabria, Italy)
Marinette Savonnet (Université de Bourgogne, France)
Jason Sawin (University of St. Thomas, St. Paul, United States)
Atsuhiko Takasu (National Institute of Informatics, Japan)
Giorgio Terracina (University of Calabria, Italy)
Motomichi Toyama (Keio University, Japan)
Goce Trajcevski (Iowa State University of Science and Technology, United States)
Irina Trubitsyna (University of Calabria, Italy)
Jeffrey Ullman (Stanford University, United States)
Domenico Ursino (Università Politecnica delle Marche, Italy)

Costas Vassilakis (University of Peloponnese, Greece)

Krishnamurthy Vidyasankar (Memorial University of Newfoundland, Canada)

Eugenio Vocaturo (University of Calabria, Italy)

Alicja Wieczorkowska (Polish-Japanese Insti. of Information Technology in Warsaw, Poland)

Carlo Zaniolo (University of California, Los Angeles, United States)

Ester Zumpano (University of Calabria, Italy)

External Reviewers

Will Knight (ConfSys.org, United States)

IDEAS 2021 - Organization

Track Organization Chairs

Jorge Bernardino, ISEC, Portugal
Jaroslav Pokorny, Charles University, Czech Rep.

Publicity Chairs

Elio Masciari, Univ. of Naples, Italy
Peter Revesz, Univ. of Nebraska-Lincoln, USA,

Tracks

Data Sharing, Security and Privacy,

Bipin C. Desai,

Software Engineering for Data Science

Juan J. Cuadrado-Gallego, **Olga Ormandjieva**.

IDEAS Steering Committee

Desai , Bipin C. (Chair)	<i>Concordia University</i>
McClatchey , Richard	<i>University of the West of England, Bristol</i>
Ng , Wilfred	<i>Hong Kong Univ. of Science and Technology</i>
Pokorny , Jaroslav	<i>Charles University</i>
Toyoma , Motomichi	<i>Keio University</i>
Ullman , Jeffrey	<i>Stanford University</i>

Sponsors

Bytepress/ConfSys.org
Concordia University,
In co:operation with ACM

SQL-like query language and referential constraints on tree-structured data

Foto N. Afrati
National Technical University of
Athens
Athens, Greece
afrati@gmail.com

Matthew Damigos
National Technical University of
Athens
Athens, Greece
mgdamig@gmail.com

Nikos Stasinopoulos
National Technical University of
Athens
Athens, Greece
nstinopoulos@gmail.com

ABSTRACT

In this paper we investigate within-record referential constraints on tree-structured data. We consider an SQL-like query language such that the one used in Dremel and we call it tree-SQL. We show how to define and process a query in tree-SQL in the presence of referential constraints. We give the semantics of tree-SQL via flattening and show how to produce equivalent semantics using the notion of tree-expansion of a query in the presence of referential constraints.

CCS CONCEPTS

• **Information systems** → **Semi-structured data**; **Query optimization**; **Database query processing**;

KEYWORDS

query processing, tree-structured data, referential constraints

ACM Reference Format:

Foto N. Afrati, Matthew Damigos, and Nikos Stasinopoulos. 2021. SQL-like query language and referential constraints on tree-structured data. In *25th International Database Engineering & Applications Symposium (IDEAS 2021)*, July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3472163.3472184>

1 INTRODUCTION

Analysis of large collections of complex data (e.g., tree-like, graph) has been made efficient thanks to the emergence of document-oriented databases (e.g., Elasticsearch, MongoDB) and data systems that combine a tree-structured data model and columnar storage, such as F1 [27], Dremel/BigQuery [21] and Apache Parquet. Relational databases have caught up by adding support for hierarchical data types (e.g., the JSONB type in PostgreSQL and struct in Hive).

In this paper, we draw inspiration from the Dremel data model [1, 3, 21, 22] and use the theoretical *tree-record data model* introduced in [2] for representing collections of tree-structured records. Tree-record data model supports identity and referential constraints which can be defined *within each* tree-record (called *within-record constraints*). Identity and referential constraints have been studied

extensively for the relational model and, later on, in the context of XML [5, 12, 15], graphs [14] and RDF data [7, 19]. Recently, identity constraints have been analyzed for JSON data models [4, 24]. Unlike relational databases and XML, where constraints are used for data validation, in this work, they are exploited to enable query answering by incorporating in the language a feature that allows to call on such constraints.

We consider an SQL-like query language, similar to the one used in Dremel, F1, Apache Drill, to query collections of tree-structured, and use flattening mechanism to transform the tree-structured records into relational records (i.e., unnest the nested and repeated fields in each record). Unlike the traditional flattening [3], which uses all the paths in the (tree) schema to generate the relational records, we use relative flattening [2] to flatten the tree-records with respect to the paths included the given query. Furthermore, we extend the relative flattening in order not only to take into account the within-record constraints during query answering, but also to arbitrarily navigate through multiple references (even in the case they form cycles in the schema).

Although the semantics defined via full flattening [2, 3] offer a natural way to interpret SQL-like queries, it is not efficient. Full flattening can expand the amount of space necessary to hold the data. Thus it is preferable for the query to be viewed (if possible) as a tree query [3]. Then we can compute the result by using embeddings of the tree query to the tree that represents the data. This is what is offered in this paper.

2 TREE-STRUCTURED RECORDS

In this section we discuss the data model.

We consider a table which consists of records, as in the relational databases, but the schema of each record is a tree instead of a list of attributes. The attributes that receive values in an instance of such a schema are the ones appearing in the leaves of the tree. We assign attributes to the internal nodes of the tree as well, which depict the structure of the record. All the attributes of the schema tree are associated with a group type which describes the structure of the subtree rooted at the specific attribute.¹ The formal definition is as follows:

A *group type* (or simply *group*) G is a complex data type defined by an ordered list of items called *attributes* or *fields* of unique names which are associated with a data type, either primitive type or group type. We associate an annotation with each field within a group type. This annotation denotes the repetition constraint. We describe details next.

¹In fact, group type could be thought of as an element in XML, an object in JSON, or as a Struct type in other data management systems (e.g., SparkSQL, Hive).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada
© 2021 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-8991-4/21/07.
<https://doi.org/10.1145/3472163.3472184>

Some attributes/fields are allowed to receive more than one values in any instance associated with this group type and this is denoted by an annotation in the field name. In fact, we have four options as concerns the *repetition* constraint of a field; they are used to specify the number of times a field is allowed to be repeated within the group of its parent and this annotates the name of the field. Formally, the repetition constraint for a field N can take one of the following values with the corresponding *annotation*:

- *required*: N is mandatory, and there is no annotation,
- *optional*: N is optional (i.e., appears 0 or 1 times) and is labeled by $N?$,
- *repeated*: N appears 0 or more times and is labeled by N^* ,
- *required and repeated*: N appears 1 or more times and is labeled by N^+ .

We denote by *repTypes* the set {required, repeated, optional, required and repeated} of repetition types.²

Now a *tree-schema* [2, 3] is formally defined as a group type with no parent. It is easy to see that this definition of a tree-schema can be actually described by a tree as follows:

Definition 2.1. (tree-schema) A *tree-schema* S of a table T is a tree with labeled nodes such that

- each non-leaf node (called *intermediate node*) is a group and its children are its attributes,
- each leaf node is associated with a primitive data type,
- each node (either intermediate or leaf) is associated with a repetition constraint in *repTypes*, and
- the root node is labeled by the name T of the table.

We define an instance of a tree-schema S . Considering a subtree s of S , we denote as *dummy s* the tree constructed from s by de-annotating all the annotated nodes of s and adding to each leaf a single child which is labeled by the *NULL*-value.

Definition 2.2. (tree-instance) Let S be a tree-schema and t be a tree that is constructed from S by recursively replacing, from top to down, each subtree s_N rooted at an annotated node $N\sigma$, where σ is an annotation, with

- either a dummy s_N or k s_N^d -subtrees, if $\sigma = *$ (**repeated**),
- either a dummy s_N or a single s_N^d -subtree, if $\sigma = ?$ (**optional**),
- k s_N^d -subtree, if $\sigma = +$ (**repeated and required**),

where $k \geq 1$ and s_N^d is constructed from s_N by de-annotating only its root. Then, for each non-*NULL* leaf N of t , we add to N a single child which is labeled by a value of type that matches the primitive type of N . The tree t is a *tree-record* of S . An instance of S , called *tree-instance*, is a multiset of tree-records.

Example 2.3. Consider the table *Booking* with schema S depicted in the Figure 1. At this stage, we ignore the r -labeled edges, which will be defined in the next section. The *Booking* table stores data related to reservations; each record in the table represents a single reservation. As we see in S , the *Booking* group includes a repeated and required *Service* field (i.e., *Service+*) whose reachability path is *Booking.Service*; i.e., each booking-record includes one or more

services booked by the customer. The *Type* field describes the service type, is mandatory (i.e., required), and takes values from the set {accommodation, transfers, excursions}. *Booking* and *Service* groups could include additional fields, such as date the reservation booked, start and end date of the service, that are ignored here due to space limitation. Figure 2 illustrates a tree-record of S .

2.1 Additional concepts

In the rest of this section we give a few useful definitions. Each non-required node is called *annotated node*. When we *de-annotate* a node, we remove the repetition symbol from its label, if it is annotated. $lb(N)$ represents the de-annotated label of a node N in a schema. Considering the nodes N_i, N_j of a schema S , such that N_j is a descendant of N_i , we denote by $N_i.N_{i+1} \dots N_j$ the path between N_i and N_j in S . The path of de-annotated labels between the root and a node N of a schema S is called **reachability path** of N . We omit a prefix in the reachability path of a node if we can still identify the node through the remaining path. In such cases, we say that the reachability path is given in a *short form*. To distinguish the complete form of a reachability path (i.e., the one given without omitting any prefix) from its short forms, we refer to the complete form as the *full* reachability path.

In the figures, all the dummy subtrees are ignored. The reachability path of a node N in a tree-record is similarly defined as the path from the root of the tree-record to N . We assume that each node of both tree-schema and tree-record has a unique virtual id.

We now define an instantiation in a multiset rather than in a set notion, which describes a mapping from the tree-schema to each tree-record in a tree-instance. Let S be a schema and t be a tree-record in an instance of S . Since each node of S is replaced by one or more nodes in t , there is at least one mapping μ , called *instantiation*, from the node ids of S to the node ids of t , such that ignoring the annotations in S , both the de-annotated labels and the reachability paths of the mapped nodes match. The subtree of t which is rooted at the node $\mu(N)$ is an *instance* of the node N , where N is a node of S . If N is a leaf, an instance of N is the single-value child of $\mu(N)$.

We say that two subtrees s_1, s_2 of a tree-record are *isomorphic* if there is a bijective mapping h from s_1 to s_2 such that the de-annotated labels of the mapped nodes match. We say that s_1 and s_2 of t are equal, denoted $s_1 = s_2$, if they are isomorphic and the ids of the mapped nodes are equal.

3 WITHIN-RECORD REFERENTIAL CONSTRAINTS

In this section, we define the concepts of within-record identity and referential constraints for a tree-schema. Due to repetition constraints, we might have fields that uniquely identify other fields (or subtrees) in each record, but not the record itself [2, 6]. In Example 2.3, each service has an identifier which is unique for each service within a reservation, but not unique across all the reservations in the *Booking* table. To support such type of constraints in a tree-record data model, we define the concept of *identity constraint with respect to a group*.

Definition 3.1. (identity constraint) Let S be a tree-schema with root R_0 , \mathcal{D} be a tree-instance of S , and N, I be nodes of S such

²Note that a repeated field can be thought of as an array of elements (repeated types) in JSON structures.

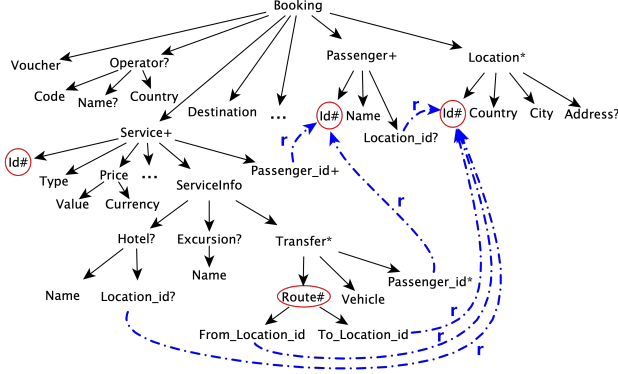


Figure 1: Booking Schema - Tree-record model with references

that N is intermediate and I is a descendant of N . Suppose that M is the parent of N , if $N \neq R_0$; otherwise, $M = N = R_0$. An *identity constraint with respect to N* is an expression of the form $I \xrightarrow{1} N$, such that I and all the descendants of I in S are required. We say that $I \xrightarrow{1} N$ is *satisfied* in \mathcal{D} if for each $t \in \mathcal{D}$ and for each instance t_p of M in t , there are not two isomorphic instances of I in t_p . I is called *identifier* and N is the *range group* of I .

In the tree representation of a schema, for each $I \xrightarrow{1} N$, we use the symbol $\#$ to annotate the identifier I (i.e., $I\#$). We also use a special, dotted edge (I, N) , called *identity edge*, to illustrate range group N of I . If N is the parent of I , we omit such an edge, for simplicity.

Example 3.2. Continuing the Example 2.3, each reservation-record includes a list of passengers which is given by the field *Passenger+*. The *Passenger* includes the following 3 fields:

- *Passenger.Id*,
- *Passenger.Name*,
- *Passenger.Location_id*,

where the last one is optional for each passenger. Notice also that there is the identity constraint $Passenger.Id \xrightarrow{1} Passenger$, which means that the field *Passenger.Id* uniquely identifies the *Passenger*. Since the range group of the *Passenger.Id* is its parent, we ignore the corresponding identity edge. In Figure 2, we can see a tree-record that satisfies this constraint, since each *Passenger* instance has a unique *Id*.

To see the impact of the range group, let us compare the following two constraints: $Route \xrightarrow{1} Transfer$ and $Route \xrightarrow{1} Service$. The field *Route* in the former case (i.e., the one illustrated in Figure 1) is a composite identifier of its parent and consists of two location ids; i.e., the combination of From and To locations uniquely identifies the transfer instances within each service, but not across all the services of the booking. On the other hand, setting the range group of the *Route* to *Service* (i.e., the latter constraint), the combination of From and To locations are unique across all the transfer services in each booking-record.

We now define the concept of *referential constraint* (or, simply *reference*), which intuitively links the values of two fields. In essence,

the concept of reference is similar to the foreign key in relational databases, but, here, it is applied within each record.

Definition 3.3. (referential constraint) Let I, N and R be nodes of a tree-schema S such that $I \xrightarrow{1} N$, R is not a descendant of N , and I, R have the same data type. A *referential constraint* is an expression of the form $R \xrightarrow{r} I$. A tree-instance \mathcal{D} of S *satisfies* the constraint $R \xrightarrow{r} I$, if for each tree-record $t \in \mathcal{D}$ the following is true: For each instance t_{LCA} of the lowest common ancestor (LCA) of R and N in t , each instance of R in t is isomorphic to an instance of I in t_{LCA} . If I is a leaf, then $R \xrightarrow{r} I$ is called *simple*.

If we have $R \xrightarrow{r} I$, we say that R (called *referrer*) *refers to* I (called *referent*). To represent the constraint $R \xrightarrow{r} I$ in a tree-schema S , we add a special (dashed) edge (I, R) , called *reference edge*, which is labeled by r . Let C be the set of identity and referential constraints over S . Consider now the tree B given by (1) ignoring all the reference and identity edges, and (2) de-annotating the identifier nodes. We say that a collection \mathcal{D} is a tree-instance of the tree-schema S in the presence of C if \mathcal{D} is a tree-instance of B and \mathcal{D} satisfies all the constraints in C .

Example 3.4. Continuing the Example 3.2, we can see that the schema S depicted in Figure 1 includes two referrers of the *Passenger.Id*; *Service.Passenger_Id* and *Transfer.Passenger_Id* store the ids of the passengers that booked each service and the ids of the passengers taking each transfer, respectively. Furthermore, the *Location.Id* is a referent in four references defined, while the composite identifier *Route* consists of two fields that both refer to the *Location.Id* identifier.

Consider now the tree-record t of S which is depicted in Figure 2. It is easy to see that this reservation includes 3 services booked for two passengers. The first service is booked for the first passenger, while the services with ids 2 and 3 are taken by both passengers. In each service, there is a list of passenger ids representing the passengers taking each service. Those fields refer to the corresponding passengers in the passenger list of the booking; appropriate reference edges illustrate the references in t .

4 SQL-LIKE QUERY LANGUAGE

We use the Select-From-Where-GroupBy expressions used in Dremel [3, 21] to query tables defined through a tree-schema. We refer to such a query language as *Tree-SQL*.

In detail, a query Q is an expression over a table \mathcal{T} with schema S and tree-instance \mathcal{D} in the presence of within-constraints C . We consider *only* simple references in C ³. The expression Q has the following form, using the conventional SQL syntax [17]:

SELECT *expr* **FROM** \mathcal{T} [**WHERE** *cond*] [**GROUP BY** *grp*],

where *cond* is a logical formula over fields of S and *grp* is a list of grouping fields. *cond*, *expr* and *grp* are defined in terms of the leaves of S . *expr* is a list of selected leaves of S followed by a list of aggregations over the leaves of S . In the case that *expr* includes both aggregated operators and fields that are not used by aggregations, those fields should be present in the *GROUP BY* clause. Each leaf

³Querying schemas having references to intermediate nodes is considered a topic for future investigation.

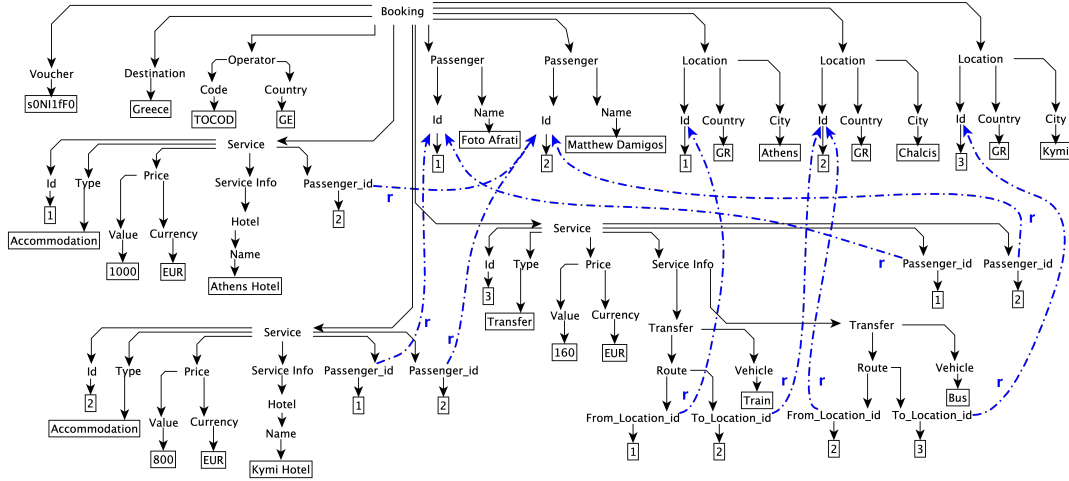


Figure 2: Booking instance - Tree-record with references

node in Q is referred through either its reachability path or a path using reference and identity edges implied by the constraints in C .

To analyze the semantics of a query in more detail, we initially ignore the references and identifiers.

We start with an example. Consider the following query Q over the table *Booking* with schema S depicted in Figure 1:

```
SELECT Voucher, Destination, Operator.Name
FROM Booking
WHERE Operator.Country='GE';
```

When Q is applied on an instance \mathcal{D} of S it results a relation, also denoted $Q(\mathcal{D})$, including a single tuple for each tree-record t in \mathcal{D} such that the instance of the *Operator.Country* field in t is 'GE'. Each tuple in $Q(\mathcal{D})$ includes the voucher of the booking, the destination and the name of the operator (if it exists - otherwise, the *NULL*-value). For example, if \mathcal{D} includes the tree-record illustrated in Figure 2, then $Q(\mathcal{D})$ includes the tuple (s0N11fF0, Greece, NULL).

We give the semantics of the language via the concept of flattening [2, 3] which is presented in Section 4.1.

Observations and discussion: Although navigation languages (e.g., XPath, XQuery, JSONPath⁴) are used to query a single tree-structured document, here, we focus on a combination of a simple navigation language and SQL-like language to query collections of tree-structured data.

In this work, we do not consider joins, recursion, nested queries and within-aggregations [21], as well as operations that are used to build a tree-like structure at query-time, or as a result of the query (e.g., the *json_build*-like functions in PostgreSQL).

In the previous example, we can see that the fields used in both *SELECT* and *WHERE* clauses do not have any repeated field in their reachability path. Querying the instances of such kind of fields is similar to querying a relation consisting of a column for each field. The tuples are constructed by assigning the single value of each field, in each record, to the corresponding column. The challenge comes up when a repeated node exists in the reachability path of

a field used in the query; since such a field might have multiple instances in each tree-record.

4.1 Flattening nested data

In this section, we analyze the flattening operation applied on tree-structured data. Flattening is a mapping applied on a tree-structured table and translates the tree-records of the table to tuples in a relation. By defining such a mapping, the semantics of Tree-SQL is given by the conventional SQL semantics over the *flattened relation* (i.e., the result of the flattening over the table). Initially, we consider a tree-schema without referential and identity constraints.

Let S be a tree-schema of a table T and \mathcal{D} is an instance of S , such that there is not any reference defined in S . Suppose also that N_1, \dots, N_m are the leaves of S . The *flattened relation* of \mathcal{D} , denoted $flatten(\mathcal{D})$, is a multiset given as follows:

$flatten(\mathcal{D}) = \{ \{ lb(\mu(N_1)), \dots, lb(\mu(N_m)) \} \mid \mu \text{ is an instantiation of a tuple } t \in \mathcal{D} \}$.

For each pair (N_i, N_j) , $\mu(N_i)$ and $\mu(N_j)$ belong to the same instance of the lowest common ancestor of N_i and N_j in t . Considering now a query Q over S and an instance \mathcal{D} of S , we say that Q is evaluated using *full flattening*, denoted $Q(flatten(\mathcal{D}))$, if $Q(\mathcal{D})$ is given by evaluating Q over the relation $flatten(\mathcal{D})$. It's worth noting here that if S does not have any repeated field then the flattened relation of \mathcal{D} includes $|\mathcal{D}|$ tuples; otherwise, each record in \mathcal{D} can produce multiple tuples during flattening.

Example 4.1. Let T be a table with schema S depicted in Figures 3(a), and instance \mathcal{D} including only the tree-record depicted in Figures 3(b). The flattened relation $flatten(\mathcal{D})$ is $\{ \{ (V_1, V_2, V_4), (V_1, V_3, V_4), (V_1, V_5, NULL) \} \}$. Consider now the query Q : *SELECT* N_1, N_4 *FROM* T . Q typically applies a projection over the flattened relation; hence, it results three tuples; i.e., $\{ (V_1, V_4), (V_1, V_4), (V_1, NULL) \}$. However, we see that N_4 in \mathcal{D} has two instances, V_4 and $NULL$. The evaluation of Q using full flattening is affected by the repetition of N_3 .

Now we motivate the definition of relative flattening with an example. Supposing the table \mathcal{T} with schema and record depicted in Figures 1 and 2, respectively, we want to find the total price

⁴JSONPath (2007). <http://goessner.net/articles/JsonPath>

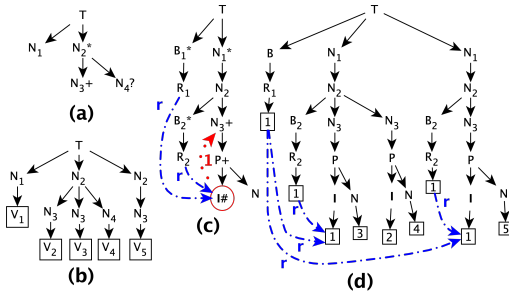


Figure 3: Flattening, out-of-range references and cycles

(i.e., $sum(Service.Price)$) for accommodation services. Although the total price is 1,800, we can see that using full flattening, the result of the query is 9,000; due to the repetition of the *Passenger* and *Location* subtrees.

To avoid cases where the repetition of a field that is not used in the query has an impact on the query result, we define the concept of *relative flattening*. Let S be a tree-schema of a table T and \mathcal{D} is an instance of S , such that there is not any reference defined in S . Consider also a query Q over T that uses a subset $L = \{N_1, \dots, N_k\}$ of the set of leaves of S , and the tree-schema S_L constructed from S by removing all the nodes except the ones included in the reachability paths of the leaves in L . Then, we say that a query Q is evaluated using *relative flattening*, denoted $Q(\text{flatten}(\mathcal{D}, Q))$, if $Q(\mathcal{D})$ is given by evaluating Q over the relation:

$flatten(\mathcal{D}, \mathcal{Q}) = \{ \{ (lb(\mu(N_1)), \dots, lb(\mu(N_k))) \mid \mu \text{ is an instantiation from the nodes of } S_L \text{ to the nodes of } t \in \mathcal{D} \} \}.$
 Continuing the Example 4.1, we have that $Q(flatten(\mathcal{D}, \mathcal{Q})) = \{ \{ (V_1, V_4), (V_1, NULL) \} \}.$

PROPOSITION 4.2. *Consider a query Q over a tree-schema S such that Q does not apply any aggregation. Then, for every instance \mathcal{D} of S , the following are true:*

- (1) *there is a tuple r in $Q(\text{flatten}(\mathcal{D}, Q))$ if and only if there is a tuple r in $Q(\text{flatten}(\mathcal{D}))$,*
- (2) $|Q(\text{flatten}(\mathcal{D}, Q))| \leq |Q(\text{flatten}(\mathcal{D}))|$.

As we discussed previously, to flatten a tree-instance with respect to a given query (i.e., using relative flattening), we use all the paths included in the query expression, regardless of whether the paths are included in the select-clause or not. Such a remark is critical, due to our familiarity with conventional SQL semantics over the relational data model. In tree-record collections, which include repeated fields, even the paths of conditions in the where-clause that are always true might significantly affect the result of a query; and specifically, the multiplicity of the tuples resulted. To see this, consider, for example, the following query Q over the schema S defined in Figure 1.

```
SELECT Voucher, Destination
FROM Booking
WHERE Service.Id = Service.Id;
```

It is easy to see that the condition in where-clause is always true. Suppose now the query Q' which is given from Q by removing the where-clause. Although a conventional SQL user would expect that $Q(\mathcal{D}) = Q'(\mathcal{D})$, for every tree-instance \mathcal{D} of S , this is not true. In particular, if $\mathcal{D} = \{\{t\}\}$, where t is the tree-record illustrated in Figure 2, we can easily see that $Q(\mathcal{D}) = \{\{(s0NI1fF0, Greece)\}\}$.

while $Q'(\mathcal{D})$ includes the tuple $(s0NI1fF0, Greece)$ 3 times; i.e., since the *Service* is a repeated field and the field *Service.Id* has 3 instances in t .

4.2 Flattening with references

In the previous section, we ignored the presence of constraints when we explained how to use flattening to answer a Tree-SQL query. Here, we show how we take advantage of the constraints to extend the query semantics based on the relative flattening.

Let us start our analysis by looking at the schema S in Figure 1. Let \mathcal{D} be an instance of S . Suppose now that we want to find, for all the transfer services in \mathcal{D} , their vouchers, along with the following transfer information: vehicle of each transfer and the route expressed as a combination of From and To cities. Note that this query cannot be answered based on the query semantics defined in the previous section⁵, since the city of each location does not belong into the same Route subtree. Taking into account, however, the following constraints, it is easy to see that intuitively such a query could be answered.

$$\begin{array}{l} \text{From_Location_id} \xrightarrow{r} \text{Location.Id} \quad \text{Location.Id} \xrightarrow{1} \text{Location} \\ \text{To_Location_id} \xrightarrow{r} \text{Location.Id} \end{array}$$

To see this, we can initially search for the voucher, vehicle, and ids of the From and To Locations for each transfer service within all the bookings. Then, for each id of the From and To Locations, we look at the corresponding Location list of the same record and identify the corresponding cities. To capture such cases and use the identity and referential constraints, we initially extend the notation of the Tree-SQL as follows. Apart from the reachability paths of the leaves that can be used in *SELECT*, *WHERE* and *GROUP BY* clauses, if there are constraints $R \xrightarrow{r} I$ and $I \xrightarrow{1} G$ over a schema S , we can use paths of the form:

$$[pathToR].R.I.G.[pathToL],$$

where the $[pathToR]$ is the full reachability path of R , L is a leaf which is a descendant of G , and $[pathToL]$ is the path of de-annotated labels from G to L in S . We call such paths (full) *reference paths*. Similarly to reachability paths, we can also use a shortened form of the reference paths by using a short form $[shortPathToR]$ of the reachability path $[pathToR]$ and omitting the nodes I , G (which are implied by the constraints); i.e., its short form is

$$[shortPathToR].R.[pathToL].$$

Hence, the query Q_{tr} answering the previous question is:

```
SELECT Voucher, Vehicle, Route.From_Location_id.City,
Route.To_Location_id.City
```

```
FROM Booking WHERE Service.Type = 'transfer';
```

Intuitively, navigating through identity and reference edges, the leaves of G become accessible from R . For example, in the schema S in Figure 1, the leaves of the group *Location* are accessible through both *From_Location_id* and *To_Location_id*.

To formally capture queries using references, we extend the relative flattening presented in the previous section as follows.

Definition 4.3. Let S be a tree-schema of a table T , \mathcal{D} be an instance of S , and C be a set of identity and referential constraints

⁵If the data is structured as in Figure 1, such a query cannot be answered using Select-From-Where-GroupBy queries in Dremel, as well.

satisfied in \mathcal{D} . Consider also a query Q over T that uses a set of leaves $\mathcal{L} = \{N_1, \dots, N_k, L_1, \dots, L_m\}$ of S so that for each L_i in \mathcal{L} there are constraints $R_i \xrightarrow{r} I_i, I_i \xrightarrow{1} G_i$ in C , where G_i is an ancestor of L_i . Suppose the construction of the following schemas from S :

- we construct a single tree-schema $S_{\mathcal{L}}$ from S by keeping only the reachability paths (without de-annotating any node) of the leaves in $\{N_1, \dots, N_k, R_1, \dots, R_m\}$, and
- for each unique G_i so that there is at least $L_i \in \mathcal{L}$, we construct a tree-schema S_{G_i} which includes only the reachability paths of R_i, I_i and the leaves of G_i that are included in $\{L_1, \dots, L_m\}$.

Both $S_{\mathcal{L}}$ and S_{G_i} keep the node ids from S . A query Q is evaluated in the presence of the constraints in C , denoted $Q(\text{flatten}(\mathcal{D}, Q, C))$, if $Q(\mathcal{D})$ is given by evaluating Q over the relation: $\text{flatten}(\mathcal{D}, Q, C) = \{\{(lb(\mu(N_1)), \dots, lb(\mu(N_k)), lb(\mu_1(L_1)), \dots, lb(\mu_m(L_m))) \mid$

- μ is an instantiation from the nodes of $S_{\mathcal{L}}$ to the nodes of $t \in \mathcal{D}$,
- each μ_i is an instantiation from the nodes of S_{G_i} to the nodes of t ,
- for every two L_i, L_j s.t. $G_i = G_j$ and $R_i = R_j$, we have that $\mu_i = \mu_j$, and
- for each i , we have that $\mu(R_i) = \mu_i(R_i)$ and $lb(\mu_i(R_i)) = lb(\mu_i(I_i))\}$.

Posing Q_{tr} (defined above) on an instance \mathcal{D} including the tree-record depicted in Figure 2, we have two instantiations, each of which maps on a different instance of *Transfer* subtree. For each such instantiation, there is a single instantiation to a *Location* instance so that the referrer value equals the *Location.Id* value. The result $Q_{tr}(\mathcal{D})$ is: $\{(s0NI1fF0, \text{Train}, \text{Athens}, \text{Chalcis}), (s0NI1fF0, \text{Bus}, \text{Chalcis}, \text{Kymi})\}$. If we replace *Route.To_Location_id.City* with the field *Location.City*, the reference *To_Location_id* \xrightarrow{r} *Location.Id* is not used; hence, the result includes 6 tuples computed by combination of the 2 cities of From-location, *Athens* and *Chalcis*, and all the available instances of the *Location.City*.

Example 4.4. Consider the table T with tree-schema S and instance $\mathcal{D} = \{t\}$, where S and t are depicted in the Figure 4. Notice that the virtual id of each node is depicted next to the node in each tree. Let Q be a query over T defined as follows:

```
SELECT N6, N5, N5.N3, N8.N3
FROM T
WHERE N0 > 2;
```

The result of the query Q when it is posed on \mathcal{D} (i.e., $Q(\mathcal{D})$) is illustrated in Table 2a (the first column is just a sequential number that is not included in the result). To compute $Q(\mathcal{D})$, we initially find the paths used in Q . These paths are summarized in Table 1, where both short and full forms of the paths (reachability and reference paths) are included; for simplicity, we omit the annotations.

Following the flattening definition presented earlier in this section, we firstly construct the schema $S_{\mathcal{L}}$ depicted in the Figure 4, where $\mathcal{L} = \{N_0, N_5, N_6, N_8\}$. The node N_8 is included in the list since $N_8.N_3$ is included in the select-clause of Q ; this path is given due to the constraints $N_8 \xrightarrow{r} N_2$ and $N_2 \xrightarrow{1} N_1$. N_5 is included in \mathcal{L} due to the paths $N_5.N_3$ and N_5 in the select-clause of Q . Then,

#	Short paths	Full paths	Path type	Clause
1	N_6	$T.N_4.N_6$	Reachability path	SELECT
2	N_5	$T.N_4.N_5$	Reachability path	SELECT
3	$N_5.N_3$	$T.N_4.N_5.N_2.N_1.N_3$	Reference path	SELECT
4	$N_8.N_3$	$T.N_7.N_8.N_2.N_1.N_3$	Reference path	SELECT
5	N_0	$T.N_0$	Reachability path	WHERE

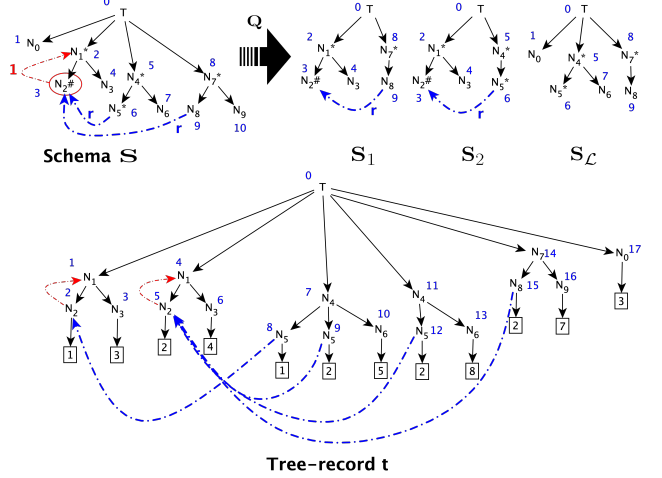
Table 1: Paths of Q 

Figure 4: Schema construction w.r.t to a query

we construct the schemas S_1 and S_2 for the referencers N_8 and N_5 , respectively, which are also depicted in Figure 4. Finding now the instantiations of the schemas S_1, S_2 and $S_{\mathcal{L}}$ over t we get the result $Q(\mathcal{D})$. Table 2b includes the instantiations for each schema that are used to compute the answer (#1) in Table 2a; e.g., the first instantiation maps the node N_0 of the schema $S_{\mathcal{L}}$ to the node with id 17 in t . Next, we join these instantiations so that the instantiation of $S_{\mathcal{L}}$ joins with the instantiations of both S_1 and S_2 on the nodes N_3 and N_5 (included in both schemas), respectively. The result is given by projecting the join result on the nodes included in the select-clause of Q , i.e., N_6, N_5, N_3 (through N_5), N_3 (through N_8), and getting their primitive values in t .

#	N_6	N_5	$N_5.N_3$	$N_8.N_3$
1	5	1	3	4
2	5	2	4	4
3	8	2	4	4

(a) Result $Q(\mathcal{D})$ of Q over t .

Schema	T	N_0	N_1^*	$N_2^\#$	N_3	N_4^*	N_5^*	N_6	N_7^*	N_8
$S_{\mathcal{L}}$	0	17				7	8	10	14	15
S_1	0		4	5	6				14	15
S_2	0		1	2	3	7	8			

(b) Instantiations over t that produce the first record in Table 2a.Table 2: Result of Q and instantiations of the schemas S_1, S_2 and $S_{\mathcal{L}}$ in Figure 4.

4.3 Well-defined referential constraints

When we define a reference constraint we should also make sure that it is well-defined in the following sense. If the reference is within-range then we say that we have a well-defined reference.

Definition 4.5. Let S be a tree-schema, and $R \xrightarrow{r} I$, $I \xrightarrow{1} G$ be two constraints over S . Let L be the lowest common ancestor of G and R . Then, we say that the reference $R \xrightarrow{r} I$ is *out-of-range* if there is at least one repeated group on the path from L to G ; otherwise, the reference is *within-range*.

In the following example, we explain what goes wrong when a reference constraint is out-of-range.

Example 4.6. Consider the tree-schema S depicted in Figure 3(c) and an instance \mathcal{D} of S including the tree-record in Figure 3(d). As we can see, there are 2 referrers, R_1 and R_2 , which both refer to the identifier I . The range group of I is the node N_3 . Consider the queries Q_1 and Q_2 selecting only the fields R_1 and R_2 , respectively. Note that the result $Q_2(\mathcal{D})$ includes two times the value 1, while $Q_1(\mathcal{D})$ includes the value 1 once. Let now Q'_1 and Q'_2 be the queries selecting the paths $R_1.N$ and $R_2.N$, respectively. We can see that both $Q'_1(\mathcal{D})$ and $Q'_2(\mathcal{D})$ include the tuples (3) and (5). Hence, when we use the reference from R_2 , the number of tuples in the result remains the same. Using however the reference from R_1 , the number of tuples in the result increases. This is because it is not clear which is the instance of I that the instance of R_1 refers to. This property is captured by the following definition and proposition.

PROPOSITION 4.7. Let C be a reference $R \xrightarrow{r} I$ over a tree-schema S , and t be a record of a tree-instance \mathcal{D} of S satisfying the constraint. If C is *within-range*, then for each instance of R in t , there is a single instance of I in t .

The property described in the Proposition 4.7 is very important for defining referential constraints, since setting up out-of-range constraints the queries using the references might not compute the “expected” results as it is demonstrated in Example 4.6.

5 ANSWERING QUERIES USING TREE-EXPANSIONS

In this section, we present an alternative method for computing the result of a query, which is based on the concepts of *tree-expansion* and *embedding*. In the next subsection, we first present this method when there is only one reference in each path. Then, we extend in Section 5.2 to multiple references.

5.1 Tree-expansion for a single reference

Let us initially define the concept of tree-expansion, which typically describes a query tree constructed from the tree-schema, the constraints and the query expression.

Definition 5.1. Let C be a set of constraints over a schema S of a table T and Q be a query over T . We consider that all the paths (both reachability and reference) in Q are given in their full form. We construct a tree Q_t from all the paths occurring in Q as follows:

- (1) Each path occurring in Q is a path of Q_t , and there is no path in Q_t that does not occur in Q .
- (2) Each node in Q_t has a distinct reachability path.
- (3) Each leaf node has a child-node labeled by a distinct variable.

The tree Q_t is called the *tree-expansion* of Q and is denoted as $Gr(Q)$.

We consider that each node of $Gr(Q)$ is associated with a unique identifier, not necessarily the same with the one given in S . Typically, the tree-expansion of a query Q over a schema S is constructed as follows. We collect all the paths of Q , in their full form, and construct a tree from these paths so that all common nodes of the paths are merged. Note here that there might be multiple, different reference paths in Q referring to the same node in S . In such a case, we do not merge the suffixes of those reference paths in $Gr(Q)$, but keep them separated⁶. $Gr(Q)$ can also be constructed from S by keeping the paths occurring in Q (and de-annotating each node), and removing all the nodes and edges of S that are not included in any path of Q .

Example 5.2. Consider the tree-schema S and the query Q over S , both introduced in Example 4.4. To construct the tree-expansion $Gr(Q)$ of Q , we initially take the full form of all the paths in Q . These paths are listed in Table 1.

Combining these paths we construct the tree-expansion $Gr(Q)$ of Q , which is illustrated in Figure 5. As you can see, the first three paths in the Table 1 have the same prefix, which is given by the sub-path $T.N_4$. These paths are merged and share the same sub-path $T.N_4$ in $Gr(Q)$. In addition, although there is a single subtree in S that is rooted at the node N_1^* , $Gr(Q)$ has two paths $N_2.N_1.N_3$, one rooted at N_5 and one rooted at N_8 . These two paths are constructed from the reference paths (#3) and (#4) in Table 1. One could say that the two paths $N_2.N_1.N_3$ could be merged. Merging however these paths, we miss records in the result as we will see later in this section.

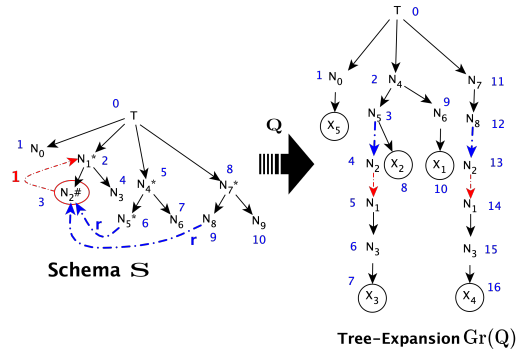


Figure 5: Tree-expansion of Q in Example 4.4.

Let us now see how the tree-expansion can be used to alternatively find the flattened relation, and consequently answer a query. To see this, we define the concept of embedding, which, in essence, describes a mapping from a tree-expansion to a tree-record.

Definition 5.3. Let Q be a query defined on a table T with schema S and instance \mathcal{D} . Suppose a mapping h from the nodes of $Gr(Q)$ to the nodes of $t \in \mathcal{D}$ so that

- for each node n of $Gr(Q)$, if n is not a variable then $lb(n) = lb(h(n))$,
- for each variable v of $Gr(Q)$, $h(v)$ is a leaf of t ,
- $h(r) = r_t$, where r and r_t are the roots of $Gr(Q)$ and t , respectively, and

⁶Otherwise, we would not have a tree but a graph.

- for each edge (n_1, n_2) of $Gr(Q)$, there is an edge $(h(n_1), h(n_2))$ in t .

Then, h is called an *embedding* from Q to t . The multiset $\{\{lb(h(X_1)), \dots, lb(h(X_k))\} \mid h \text{ is an embedding from } Q \text{ to a tree-record } t \in \mathcal{D} \text{ and } X_1, \dots, X_k \text{ are the variables of } Gr(Q)\}\}$ is called the *relational result* of $Gr(Q)$ over \mathcal{D} and is denoted as $Gr(Q, \mathcal{D}, C)$.

The following theorem shows that evaluating Q over the relational result of $Gr(Q)$ equals the evaluation of Q over the flattened relation.

THEOREM 5.4. *Let Q be a query over a table with schema S and a set of constraints C over S . Then, for each instance \mathcal{D} of S we have that $flatten(Q, \mathcal{D}, C) = Gr(Q, \mathcal{D}, C)$ and $Q(\mathcal{D}) = Q(flatten(Q, \mathcal{D}, C)) = Q(Gr(Q, \mathcal{D}, C))$.*

Example 5.5. Continuing the Example 5.2, let us see how the result of Q described in Example 4.4 is computed using the tree-expansion of Q . Here, we focus on describing how the first answer included in Table 2a is computed.

We recall that the tree-expansion $Gr(Q)$ of Q is depicted in the Figure 5. Computing now the embeddings from $Gr(Q)$ to t , we are able to find the embedding h illustrated in Figure 6 (for simplicity, we illustrate only the mapping of the nodes included in the SELECT-clause), where the root of $Gr(Q)$ maps to the root of t (i.e., $h(0) = 0$), and

$$\begin{array}{llll} h(1) = 17, & h(4) = 2, & h(9) = 10, & h(13) = 5, \\ h(2) = 7, & h(5) = 1, & h(11) = 14, & h(14) = 4, \\ h(3) = 8, & h(6) = 3, & h(12) = 15, & h(15) = 6. \end{array}$$

As we can see, the path $T.N_4.N_5.N_2.N_1.N_3$ (i.e., 0.2.3.4.5.6, in terms of node ids) of $Gr(Q)$ maps on the path 0.7.8.2.1.3 of t (given in terms of node ids), where the edge (8, 2) is a reference edge and the edge (2, 1) is an identity edge. Similarly, the path 0.11.12.13.14.15 (i.e., $T.N_7.N_8.N_2.N_1.N_3$) of $Gr(Q)$, given using node ids, maps on the path 0.14.15.5.4.6 of t , where the edge (15, 5) is a reference edge and the edge (5, 4) is an identity edge. Using the embedding h , the tuple $(X_1, X_2, X_3, X_4, X_5)$ maps on the tuple of primitive values (5, 1, 3, 4, 3), which is the same answer computed in Example 4.4 and gives the first answer in Table 2a. In Figure 6, the primitive values in t that are mapped by the variables of the $Gr(Q)$ and included into the relational result of $Gr(Q)$ are circled.

It's worth noting, here, that tree-expansion could not include multiple paths with identical labels (see condition (2) in Definition 5.1); i.e., even if a single path appears multiple times in the query, these paths give a single path in the tree-expansion. Consequently, querying the instances of a repeated field so that these instances are included in different columns is not supported by the semantics presented in this paper. For example, we cannot ask the table T with schema S and instance $\mathcal{D} = \{t\}$, where t and S are depicted in Figure 5, for pairs of N_5 values (i.e., posing a query that results the tuple (1, 2), which corresponds to the pair of nodes 8 and 9 in t). Notice also that tree-expansion results, through embeddings, not only the primitive values mapped by the paths included in the select-clause but also the values mapped by the paths included in the where-clause. The latter type of values (e.g., the values mapped by node N_0 in Example 5.5) ensure that the conditions included in the where-clause can be checked when the

query is evaluated over the flattened relation (i.e., the relational result of the tree-expansion). Note that finding an optimized flattening and query evaluation algorithm, which could validate the where-clause conditions during flattening, is considered a topic for future investigation.

5.2 Tree-expansion for multiple references

In this section, we show how we use tree-expansions to extent the query semantics and navigate through multiple references, and cycles of references. In particular, since multiple references can be defined in a single schema, there might be paths in the schema given by navigating more than one path. For instance, looking at the schema S depicted in Figure 1, we can see that there is a path P from root node *Booking* to *Location*. *Country* through the following path (we omit, here, the annotations for simplicity):

Service.Passenger_id.Id.Passenger.Location_id.Id.Location.

The path P uses two references, hence, it is not allowed to be used in a query, according to what we have discussed in the previous sections. Even if it was allowed, flattening a query which uses such a path cannot be handled by the flattening approach given in Section 4.2. Let us now see an additional, complex example, where the tree-schema includes cycles of references.

Example 5.6. Consider the table *Dept* with tree-schema S and instance $\mathcal{D} = \{t\}$, where both S and t are illustrated in Figure 7. The table *Dept* stores the projects (*Proj*) of each department, along with the employees (*Empl*) of the department. Each project has a number of employees working on it, and each employee of the department might be accountable for (*AccFor*) a list of projects. Hence, it is easy to see that the references defined between *Proj* and *Empl* subtrees form a cycle of references. One could ask for the projects of a certain category (e.g., *Cat2* = 'Cat2') which employ an employee who is accountable for a project of a different category, along with the name of the employee. To answer such a question, we need to navigate through both links. The query semantics defined in Section 4.2 allow only a single use of a reference between two subtrees.

To handle such cases, we initially extend the paths supported in query expressions. In particular, considering a tree-schema S and the constraints $R_j \xrightarrow{r} I_j$ and $I_j \xrightarrow{l} G_j$ over S , with $j = 1, \dots, n$, which are included in a set C , we generalize the reference paths to be of the following (full) form:

$[p(R, R_1)].R_1.I_1.G_1.[p(G_1, L_1)].R_2.I_2.G_2.[p(G_1, L_1)].\dots.R_n.I_n.G_n.[p(G_n, L_n)]$, where R is the root of S , $p(A, B)$ is the de-annotated path in S from the node A to the node B , and L_j is a descendant of G_j . To distinguish these reference paths from the paths that use a single reference (defined in Section 4.2), we refer to the former as *generalized reference paths* and to the latter as *simple reference paths*. Similarly to the simple reference paths, we also consider a shortened form of the generalized ones by omitting a prefix of the reachability path of R_1 (i.e., $p(R, R_1)$) and all the nodes I_j, G_j , with $j = 1, \dots, n$.

Suppose a schema S , a set of constraints C over S , and an instance \mathcal{D} of S . To compute now a query Q over S that uses generalized reference paths, we simply use the tree-expansion and find the

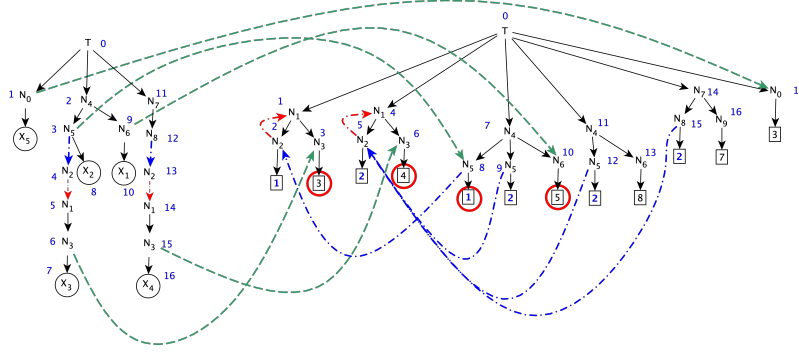
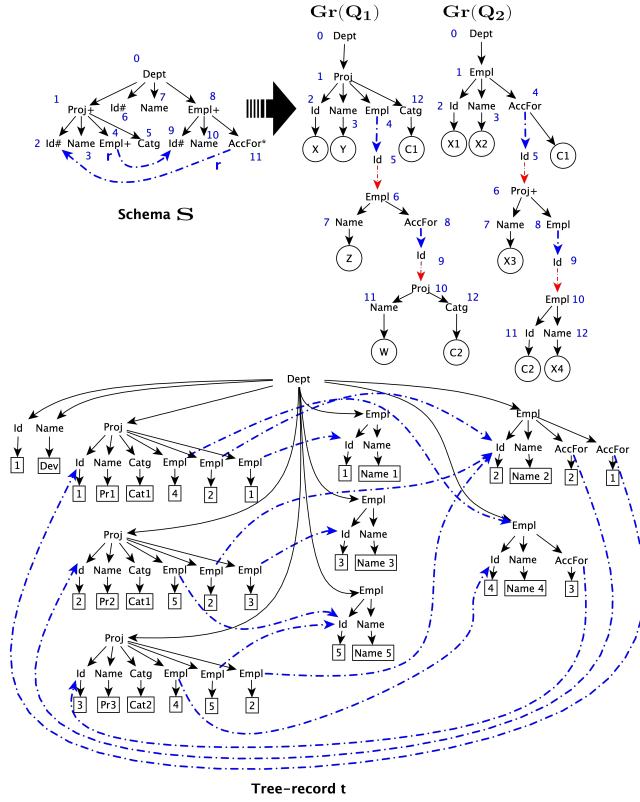


Figure 6: Embedding of a tree-expansion with a single reference in each path.

Figure 7: Schema with a cycle of references and the tree-expansions of the queries Q_1 and Q_2 defined in Example 5.6.

embeddings from the tree-expansion over each tree-record. In particular, since the definition of the tree-expansion supports generalized references paths, we construct the tree-expansion $Gr(Q)$ of Q and evaluate Q over its relational result $Gr(Q, \mathcal{D}, C)$; i.e., $Q(\mathcal{D}) = Q(Gr(Q, \mathcal{D}, C))$.

Example 5.7. Using the generalized reference paths, the question posted in Example 5.6 (for the category $Catg = 'Cat2'$) can be answered by the following query Q_1^7 :

⁷Note that the alias of the columns are used as in conventional SQL.

```
SELECT Proj.Id as X, Proj.Name as Y, Proj.Empl.Name as Z
       Proj.Empl.AccFor.Proj.Name as W
FROM Dept
WHERE Proj.Catg = 'Cat2' AND
       Proj.Empl.AccFor.Proj.Catg <> 'Cat2';
```

To construct now the tree-expansion, we initially collect the full form of all the paths used in the query Q_1 . These paths are listed in Table 3a. Notice that the path (#3) is a simple reference path that uses a single reference, while the paths (#4) and (#6) are generalized reference paths that go through two references. Combining these paths, we construct the tree-expansion $Gr(Q_1)$ of Q_1 which is depicted in Figure 7. Computing now the embeddings of $Gr(Q_1)$ over t , we have the resulting relation presented in Table 3b.

#	Full paths	Path type	Clause
1	Dept.Proj.Id	Reachability	SELECT
2	Dept.Proj.Name	Reachability	SELECT
3	Dept.Proj.Empl.Id.Empl.Name	Reference	SELECT
4	Dept.Proj.Empl.Id.Empl.AccFor.Id.Proj.Name	Reference	SELECT
5	Dept.Proj.Catg	Reachability	WHERE
6	Dept.Proj.Empl.Id.Empl.AccFor.Id.Proj.Catg	Reference	WHERE

(a) Paths of Q_1 .

#	X	Y	Z	W
1	3	Pr3	Name 2	Pr2
2	3	Pr3	Name 2	Pr1

(b) Result $Q_1(\mathcal{D})$ of Q_1 over t .Table 3: Evaluating Q_1 over t .

Let us now try to find the employees who are accountable for a project, the project they are accountable for, and the employees working on the certain project. To answer this question, we need to start our paths from the *Empl* field and go through projects, in order to find the projects each employee is accountable for, and then navigate to the employees of each such project to find the team members. In particular, the query Q_2 answering this question is given as follows:

```
SELECT Empl.Id as X1, Empl.Name as X2,
       Empl.AccFor.Proj.Name as X3,
       Empl.AccFor.Proj.Empl.Name as X4
FROM Dept
WHERE Empl.AccFor is not NULL AND
       Empl.Id <> Empl.AccFor.Proj.Empl.Id;
```

To answer the query Q_2 , we construct the tree-expansion $Gr(Q_2)$ which is depicted in Figure 7. The result $Q_2(\mathcal{D})$ is give in Table 4.

As we can see in the tree-expansions of Q_1 and Q_2 in Example 5.7, we start unfolding the paths from a different field (from the *Proj* field in $Gr(Q_1)$ and from the *Empl* for $Gr(Q_2)$). Typically, in cases that we have cycles of references, we can use a path of arbitrary

#	X1	X2	X3	X4
1	2	Name 2	Pr1	Name 4
2	2	Name 2	Pr1	Name 1
3	2	Name 2	Pr2	Name 3
4	2	Name 2	Pr2	Name 5
5	4	Name 4	Pr3	Name 2
6	4	Name 4	Pr3	Name 5

Table 4: Result of Q_2 over t .

length in the query, navigating through the paths multiple times. These paths, however, should be clearly specified in the query; i.e., we do not consider, here, any recursive operator as in XPath.

6 RELATED WORK

Work on constraints for tree-structured data has been done during the past two decades. Our work, as regards the formalism, is closer to [5, 6, 28, 29]. The papers [6] and [5] are among the first works on defining constraints on tree-structured data. Reasoning about keys for XML is done in [6] where a single document XML is considered and keys within scope (relative keys) are introduced. Referential constraints through inclusion dependencies are also investigated (via path expression containment). The satisfiability problem is investigated, but no query language is considered. Many recent works investigate discovering conditional functional dependencies in XML Data; closer to our perspective is [28] and [29] where XML schema refinement is studied through redundancy detection and normalization.

[24] and [4] focus on the JSON data model and a similar to XPath navigational query language. These works also formalize specification of unique fields and references, they do not define relative keys. [24] formally defines a JSON schema. It supports specification of unique fields within an object/element and supports references to an another subschema (same subschema can be used in several parts of the schema). No relative keys are supported. [4] continues on [24] and proposes a navigational query language over a single JSON document (this language presents XPath-like alternatives for JSON documents, such as JSONPath, MongoDB navigation expressions and JSONiq).

Flattening has initially been studied in the context of nested relations and hierarchical model (e.g., [8, 23, 26]). Dremel [3, 21, 22], F1 [27] and Drill use flattening to answer SQL-like queries over tree-structured data. Flattening semi-structured data is also investigated in [10, 11, 20], where the main problem is to translate semi-structured data into multiple relational tables.

7 FUTURE WORK

Towards future work, we want to use tree-expansion of the query to distinguish cases where the output of the query can also be given as tree-structured data (because not every flattened data can be unflattened to a tree structure). Whenever this is possible it may enable a shorter form of flattening, called semi-flattening which can be used in conjunction of columnar storage to answer queries efficiently [3]. In addition, we plan to investigate querying tree-schemas having references to intermediate nodes. Also, we aim to study flattening when the referrer is defined in the range group of the referent. Furthermore, we plan to extend this investigation towards the following directions: a) Study the satisfiability and the implication problems for the constraints we defined here. b) The

chase [25] is used to reason about keys and functional dependencies. For relational data, there is a lot of work on chase. The chase for RDF and graph data was studied in [18], [13, 16], [9] and [14]. We plan to define a new chase that can be applied to reason about the constraints we defined here.

REFERENCES

- [1] Foto N. Afrati and Rada Chirkova. 2019. *Answering Queries Using Views, Second Edition*. Morgan & Claypool Publishers.
- [2] Foto N. Afrati and Matthew Damigos. 2021. Querying collections of tree-structured records in the presence of within-record referential constraints. In *DEXA*.
- [3] Foto N. Afrati, Dan Delorey, Mosha Pasumansky, and Jeffrey D. Ullman. 2014. Storing and querying tree-structured records in dremel. *Vldb Endow.* (2014).
- [4] Pierre Bourhis, Juan L. Reutter, Fernando Suárez, and Domagoj Vrgoc. 2017. JSON: Data model, Query languages and Schema specification. In *PODS*.
- [5] Peter Buneman, Susan B. Davidson, Wenfei Fan, Carmem S. Hara, and Wang Chiew Tan. 2002. Keys for XML. *Comput. Networks* (2002).
- [6] Peter Buneman, Susan B. Davidson, Wenfei Fan, Carmem S. Hara, and Wang Chiew Tan. 2003. Reasoning about keys for XML. *Inf. Syst.* (2003).
- [7] Diego Calvanese, Wolfgang Fischl, Reinhard Pichler, Emanuel Sallinger, and Mantas Simkus. 2014. Capturing Relational Schemas and Functional Dependencies in RDFS. In *AAAI*.
- [8] Latha S. Colby. 1989. A Recursive Algebra and Query Optimization for Nested Relations. In *SIGMOD*.
- [9] Alvaro Cortés-Calabuig and Jan Paredaens. 2012. Semantics of Constraints in RDFS. In *AMW 2012*.
- [10] Alin Deutsch, Mary F. Fernández, and Dan Suciu. 1999. Storing Semistructured Data with STORED. In *SIGMOD*.
- [11] Michael DiScala and Daniel J. Abadi. 2016. Automatic Generation of Normalized Relational Schemas from Nested Key-Value Data. In *SIGMOD*.
- [12] Wenfei Fan. 2005. XML Constraints: Specification, Analysis, and Applications. In *DEXA*.
- [13] Wenfei Fan, Zhe Fan, Chao Tian, and Xin Luna Dong. 2015. Keys for Graphs. *Vldb Endow.* (2015).
- [14] Wenfei Fan and Ping Lu. 2019. Dependencies for Graphs. *ACM Trans. Database Syst.* (2019).
- [15] Wenfei Fan and Jérôme Siméon. 2003. Integrity constraints for XML. *J. Comput. Syst. Sci.* (2003).
- [16] Wenfei Fan, Yinghui Wu, and Jingbo Xu. 2016. Functional Dependencies for Graphs. In *SIGMOD*.
- [17] Hector Garcia-Molina, Jeffrey D. Ullman, and Jennifer Widom. 2009. *Database systems - the complete book (2. ed.)*. Pearson Education.
- [18] Jelle Hellings, Marc Gyssens, Jan Paredaens, and Yuqing Wu. 2016. Implication and axiomatization of functional and constant constraints. *Ann. Math. Artif. Intell.* (2016).
- [19] Georg Lausen, Michael Meier, and Michael Schmidt. 2008. SPARQLing constraints for RDF. In *EDBT*.
- [20] Zhen Hua Liu, Beda Christoph Hammerschmidt, and Douglas McMahon. 2014. JSON data management: supporting schema-less development in RDBMS. In *SIGMOD*.
- [21] Sergey Melnik, Andrey Gubarev, Jing Jing Long, Geoffrey Romer, Shiva Shivakumar, Matt Tolton, and Theo Vassilakis. 2010. Dremel: interactive analysis of web-scale datasets. *Vldb Endow.* (2010).
- [22] Sergey Melnik, Andrey Gubarev, Jing Jing Long, Geoffrey Romer, Shiva Shivakumar, Matt Tolton, Theo Vassilakis, Hossein Ahmadi, Dan Delorey, Slava Min, Mosha Pasumansky, and Jeff Shute. 2020. Dremel: A Decade of Interactive SQL Analysis at Web Scale. *Vldb Endow.* 13, 12 (2020), 3461–3472.
- [23] Jan Paredaens and Dirk Van Gucht. 1988. Possibilities and limitations of using flat operators in nested algebra expressions. In *PODS*.
- [24] Felipe Pezoa, Juan L. Reutter, Fernando Suárez, Martín Ugarte, and Domagoj Vrgoc. 2016. Foundations of JSON Schema. In *WWW*.
- [25] Fereidoon Sadri and Jeffrey D. Ullman. 1980. The Interaction between Functional Dependencies and Template Dependencies. In *SIGMOD*.
- [26] Marc H. Scholl, H.-Bernhard Paul, Hans-Jörg Schek, et al. 1987. Supporting Flat Relations by a Nested Relational Kernel. In *Vldb*.
- [27] Jeff Shute, Radek Vingralek, Bart Samwel, Ben Handy, Chad Whipkey, Eric Rollins, Mircea Oancea, Kyle Littlefield, David Menestrina, Stephan Ellner, John Cieslewicz, Ian Rae, Traian Stancescu, and Himani Apte. 2013. F1: A Distributed SQL Database That Scales. *Vldb Endow.* (2013).
- [28] Loan T. H. Vo, Jinli Cao, and J. Wenny Rahayu. 2011. Discovering Conditional Functional Dependencies in XML Data. In *ADC 2011*.
- [29] Cong Yu and H. V. Jagadish. 2008. XML schema refinement through redundancy detection and normalization. *Vldb J.* (2008).

A Framework for Enhancing Deep Learning Based Recommender Systems with Knowledge Graphs

Sudhir P. Mudur**

Serguei A. Mokhov

Yuhao Mao

mudur@cse.concordia.ca

mokhov@cs.concordia.ca

maoyuhao92@gmail.com

Computer Science and Software Engineering, Concordia University
Montreal, Quebec, Canada

ABSTRACT

Recommendation methods fall into three major categories, content based filtering, collaborative filtering and deep learning based. Information about products and the preferences of earlier users are used in an unsupervised manner to create models which help make personalized recommendations to a specific new user. The more information we provide to these methods, the more likely it is that they yield better recommendations. Deep learning based methods are relatively recent, and are generally more robust to noise and missing information. This is because deep learning models can be trained even when some of the information records have partial information. Knowledge graphs represent the current trend in recording information in the form of relations between entities, and can provide any available information about products and users. This information is used to train the recommendation model. In this work, we present a new generic recommender systems framework, that integrates knowledge graphs into the recommendation pipeline. We describe its design and implementation, and then show through experiments, how such a framework can be specialized, taking the domain of movies as an example, and the resulting improvements in recommendations made possible by using all the information obtained using knowledge graphs. Our framework, to be made publicly available, supports different knowledge graph representation formats, and facilitates format conversion, merging and information extraction needed for training recommendation models.

CCS CONCEPTS

• **Information Systems** → **Recommender Systems**; Semantic web description languages; • **Computing Methodologies** → Neural Networks; • **Software and its engineering** → Development frameworks and environments.

*Authors contributed equally to this research.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](https://permissions.acm.org).

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Association for Computing Machinery.

ACM ISBN 978-1-4503-8991-4/21/07...\$15.00

<https://doi.org/10.1145/3472163.3472183>

KEYWORDS

recommender system, framework, deep learning based recommendations, knowledge graphs, recommendation model training

ACM Reference Format:

Sudhir P. Mudur, Serguei A. Mokhov, and Yuhao Mao. 2021. A Framework for Enhancing Deep Learning Based Recommender Systems with Knowledge Graphs. In *25th International Database Engineering & Applications Symposium (IDEAS 2021)*, July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3472163.3472183>

1 INTRODUCTION

Propelled by the current pandemic, we start to see a very steep rise in the use of the Internet to support different human activities. There are more and more product/service offerings and providers on the Internet. While user acceptance has jumped, information overload has become a major problem for most Internet users. And this is where recommender system comes into play [9]. The recommender system is essentially programmatic support to significantly trim the information of interest to a user from the massive amount of information available on the Internet [3]. Applications of recommender systems are very wide. According to reports, recommender systems have brought 35% of sales revenue to Amazon [1] and up to 75% of consumption to Netflix [2], and 60% of the browsing on the Youtube homepage comes from recommendation services [4]. It is also widely used by various Internet companies. A recommender system is useful as long as there are a large number of items to offer to the clients [12, 17, 18]. The current application domains of recommender systems have transcended beyond e-commerce, into news, video, music, dating, health, education, etc.

A recommender system is essentially an information filtering system. It "learns" the users' interests and preferences based on historical behaviour of items and users, and predicts for any specific user the rating or preference for a given specific item, based on information about the item, and the user. Clearly, the more information, the recommendation method has about users and products, the better is its ability to predict. Given the vast amount of unstructured, often noisy, redundant and incomplete information available on the Internet, convenient ways to gather, structure and provide such information is the major challenge addressed in this work. Specifically, we present a new generic software framework which enables easy integration of any available information about items

and users, by including such information into a knowledge graph to be used for training the recommendation method.

In most recommendation scenarios, items may have a lot of associated knowledge in the form of interlinked information, and the network structure that depicts this knowledge is called a knowledge graph. Information in a knowledge graph is encoded in a data structure called triples made of subject-predicate-object relations. A knowledge graph greatly increases information about the item, strengthens the connection between items, provides a rich reference value for the recommendation, and can bring additional diversity and interpretability to the recommendation result.

In our opinion, we need a general framework, which (i) integrates search and update of information, (ii) enables crawling of websites for additional information, (iii) supports storing of the information in a structured, easily accessible manner, (iv) enables easy retrieval of the information about items and users as input for the training of the recommender model. Adding a knowledge graph into the recommendation framework can help us better manage knowledge data, process data, and query the information we need faster. For one, knowledge graphs as a form of structured human knowledge have drawn great research attention from both the academia and the industry [14, 21, 25]. A knowledge graph is a structured representation of facts, consisting of entities, relationships, and semantic descriptions. Knowledge graphs can be used wherever there is a relationship. Knowledge graphs have successfully captured a large number of users, including Walmart, Google, LinkedIn, Adidas, HP, FT Financial Times, etc. Applications continue to grow.

Compared with traditional data bases and information retrieval methods, the advantages of a knowledge graph are the following:

- Strong ability to express relationships: Based on graph theory and probability graph models, it can handle complex and diverse association analyses.
- Knowledge learning: it can support learning functions based on interactive actions such as reasoning, error correction, and annotation, and continuously accumulates knowledge logic and models, improves system intelligence.
- High-speed feedback: Schematic data storage method enables fast data retrieval speeds.

Knowledge graphs usually have two main types of storage formats [10, 26]: one is RDF (Resource Description Framework) based storage, and the other is graph database (e.e., neo4j), with their advantages and disadvantages.

Secondly, recommender systems have become a relatively independent research direction. This is generally considered to have started with the GroupLens system launched by the GroupLens research group of the University of Minnesota in 1994 [15]. As a highly readable external knowledge carrier, knowledge graphs provide a great possibility to improve algorithm interpretation capabilities [16]. Therefore, we are using knowledge graphs to enhance recommendation methods in this work.

Our main contribution is the following:

- We present an overall architecture that allows users to build knowledge graphs, display knowledge graphs, and enable recommender algorithms to be trained with knowledge extracted from knowledge graphs, in a domain agnostic manner. We demonstrate its application to movie recommendations.

Other contributions include:

- A pipeline architecture that allows users to crawl data, build and merge knowledge graphs in different formats, to display knowledge graphs and extract information, without having to know the underlying format details.
- A platform for recommender systems researchers to carry out experiments on top of TensorFlow and Keras.

The rest of the paper is organized as follows. We provide a brief review of existing recommendation software frameworks. Then, we describe the design and implementation of our generic framework in a top-down manner, along with examples of instantiation and specific applications. We conclude along with future extensions.

2 RELATED WORK

Literature scan reveals only a few attempts at development of frameworks for recommender systems, as most efforts have been in algorithms and methods. Below is a brief review of related work on recommendation software frameworks. We first describe these works, then provide a summary table of their main characteristics and limitations.

Yue et al. [22] in their work use gradient descent to learn the user's model for the recommendation. Three machine learning algorithms (including logistic regression, gradient boosting decision tree and matrix decomposition) are supported. Although gradient boosting decision tree can prevent overfitting and has strong interpretability, it is not suitable for high-dimensional sparse features, usually the case with items and users. If there are many features, each regression tree will consume a lot of time.

Raccoon [8] is a recommender system framework based on collaborative filtering. The system uses k-nearest-neighbours to classify data. Raccoon needs to calculate the similarity of users or items. The original implementation of Raccoon uses the well known Pearson distance which is good for measuring similarity of discrete values in a small range. But to make the calculation faster, one can also use Jaccard distance, which provides a binary rating data (ie like/dislike). But collaborative filtering algorithm does not care about the detailed features of users or products. It only uses the user ID and product ID to make recommendations.

GER (Good Enough Recommendation) [7] is presented as a scalable, easy-to-use and easy-to-integrate recommendation engine. Its core is the same as the knowledge graph triplet (people, actions, things). GER recommends in two ways. One is by comparing two people, looking at their history, and another one is from a person's history. Its core logic is implemented in an abstraction called the Event Storage Manager (ESM). Data can be stored in memory ESM or PostgreSQL ESM. It also provides corresponding interfaces to the framework developer, including (i) Initialization API for operating namespace, (ii) Events API for operating on triples, (iii) Thing Recommendations API for computing things, (iv) Person Recommendations API for recommending users, and (v) Compacting API for compressing items.

LensKit [6], is a Java open-source recommender system, produced by the GroupLens Research team at University of Minnesota. The java version has been deprecated. The new python version of Lenskit is a set of tools for experimenting and researching recommender systems. It provides support for training, running and

evaluating recommender systems. The recommendation algorithms in LensKit include SVD (singular value decomposition), Hierarchical Poisson Factorization, and KNN (K-nearest neighbors). LensKit can work with any data in pandas.DataFrame (Python Data Analysis Library) with the expected (fixed set) columns. LensKit loads data through the dataLoader function. Each data set class or function takes a path parameter specifying the location of the data set. These data files have normalized column names to fit with LensKit’s general conventions.

DKN (Deep Knowledge-Aware Network for News Recommendation) [23] proposes a model that integrates the embedded representation of knowledge graph entities with neural networks for news recommendation. News is characterized through a highly condensed representation and contains many knowledge entities, and it is time sensitive. A good news recommendation algorithm should be able to make corresponding changes as users’ interests change. To solve the above problems, the DKN model has been proposed. First, a knowledge-aware convolutional neural network (KCNN) is used to integrate the semantic representation of news with the knowledge representation to form a new embedding, and then the attention from the user’s news click history to the candidate news is established. The news with higher score is recommended.

Multi-task Feature Learning for Knowledge Graph enhanced Recommendation (MKR) [24] is a model that uses knowledge graph embedding tasks to assist recommendation tasks. These two tasks are linked by a cross-compression unit, which automatically shares potential features, and learns the high-level interaction between the recommender system and entities in the knowledge graph. It is shown that the cross-compression unit has sufficient polynomial approximation ability, and MKR is a model of many typical recommender systems and multi-task learning methods. Our framework incorporates a modified version of the MKR model.

Main Characteristics of Earlier Recommender Systems:

In the above frameworks, usually just one of the three types of recommender system models (collaborative filtering, content-based and machine learning) is supported. Table 1 summarizes the characteristics of the above frameworks.

As we can see there is no generic recommender system software framework yet that can support web crawling, information update, visualization and input to recommendation methods independent of storage formats and algorithms. We will briefly discuss the limitations of presently available frameworks from this viewpoint of genericity. Yue et al. mainly use the boosting model, which makes the training time high. Because Raccoon uses the K-nearest-neighbours model, it cannot handle new users or new items, well known as the cold start problem. Further, it also has data sparseness and scalability issues. The advantage of GER is that it contains no business rules, with limited configuration, and almost no setup required. But this is at the expense of the scalability of the engine. Other limitations are that it does not generate recommendations for a person with less history, and has the data set compression limit problem, i.e., certain items will never be used. For example, if items are older or belong to users with a long history, these items will not be used in any calculations but will occupy space. The advantage of LensKit is that its framework contains many algorithms, such as funksvd and KNN, and users can call different algorithms according to their needs. In LensKit the data is called through the

path parameter, and the data has a specific format. This means that the LensKit framework can only process user, item and rating data. If the user has new data, such as user information or item classification, the LensKit framework cannot handle it. Although DKN uses knowledge graph embedding as an aid, it is tailored to news recommendations and cannot be easily extended to other product domains. The main disadvantage of MKR is that it is not a generic framework, and it inputs text documents as knowledge graphs, but not in their graph structured form, making it cumbersome to update knowledge.

3 OUR FRAMEWORK DESIGN

Figure 1 shows the overall design of our framework. It is designed as a pipeline of tasks from end user input to final recommendations. Further, each stage in the pipeline is designed as a sub-framework, some stages are nested, enabling specialization and expansion at a more granular level. We denote a generic component as a frozen spot and its specialized component as a hot spot.

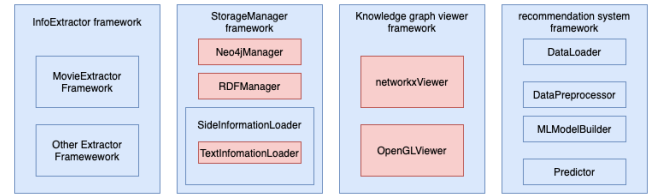


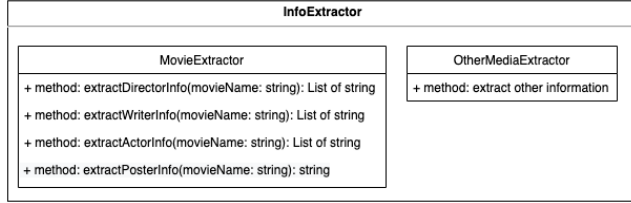
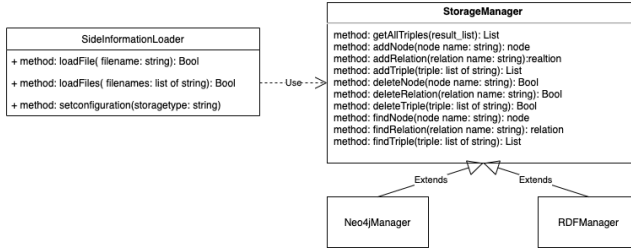
Figure 1: Core components of the framework, each box in blue means the framework’s frozen spot and each box in red represents a set of the hot spots for each specialized framework.

The four major stages in the pipeline have the following functionality:

- **InfoExtractor framework:** abstracts a web extractor. It takes a URL address (such as .html, .htm, .asp, .aspx, .php, .jsp, .jspx) and extraction rules as input then formats the extracted output. This is further nested, to enable domain level (movies, news, etc.) specialization.
- **StorageManager framework:** abstracts different knowledge graph storage formats. It takes string data stream as input and generates the output in the required format.
- **Knowledge graph viewer framework:** abstracts knowledge graph visualization. It takes the triples stream as input then creates the visualization.
- **Recommendation method:** abstracts the recommendation method and knowledge input. Knowledge graph triples form the input for training the recommendation model.

InfoExtractor Framework Design: abstracts common methods of information extractors. We take the example of extracting movie information - MovieExtractor nested in InfoExtractor. It serves as a frozen spot to provide users with functions for capturing movie information, as shown in Figure 2. MovieExtractor is the abstract class of the extractor, it has three basic methods, extractDirectorInfo, extractWriterInfo and

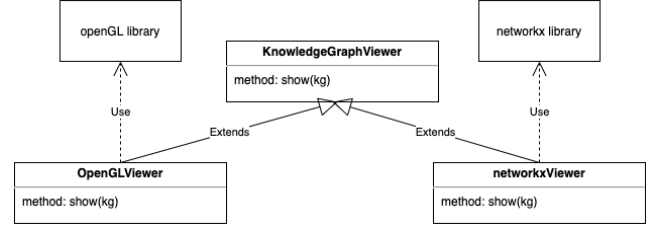
Name	Type	Storage types	Domain	Example uses	Language	ML Models	Maintained
Yue Ning et. al.	Decision Tree	-	content recommendation	content recommendation	-	Boosting	-
Racon	Collaborative Filtering	Redis	cross-domain	https://github.com/guymorita/benchmark_racon_movies	Java	KNN	Jan 10, 2017
GER	Collaborative Filtering	PostgreSQL	movie	https://github.com/grahamjenson/ger/tree/master/examples	Java	-	Jul 9, 2015
LensKit	Machine Learning	LocalFile	cross-domain	https://github.com/lenskit/lenskit/tree/master/examples	Python	SVD, hpf	Nov 10, 2020
DKN	knowledge graph enhanced	LocalFile	News	https://github.com/hwwang55/DKN	Python	tensorflow	Nov 22, 2019
MKR	knowledge graph enhanced	LocalFile	cross-domain	https://github.com/hwwang55/MKR	Python	tensorflow	Nov 22, 2019
PredictionIO	machine learning	Hadoop, HBase	-	https://github.com/apache/predictionio	Scala	Apache Spark MLlib	Mar 11, 2019
Surprise	Collaborative Filtering	LocalFile	movie, joke	https://github.com/NicolasHug/Surprise/tree/master/examples	Python	matrix factorization, KNN	Aug 6, 2020

Table 1: Characteristics of Existing Frameworks**Figure 2: Design of the InfoExtractor Framework.****Figure 3: Design of the knowledge graph StorageManager framework.**

extractActorInfo. It is used to extract director information, author information and actor information respectively. extractPosterInfo is the abstract class of the poster downloader. It downloads the movie poster (an example of side information) and then converts the image into a string, to facilitate storage and coding. There are many websites on the Internet that store information about movies, such as IMDB, Wikipedia, Netflix and Douban. Extractors dedicated to these websites can be used as hotspots.

StorageManager Framework Design: StorageManager is a frozen spot, as shown in Figure 3. Its design allows us to add support for different kinds of storage modes easily. SideInformationLoader, is again designed as a frozen spot, and its responsibility is to add triples to the knowledge graph through the method in StorageManager to increase the knowledge provided to the recommendation method. SideInformationLoader contains three methods, they are loadFile, loadFiles and setConfiguration.

Knowledge Graph Viewer Framework Design: This frozen spot responsibility is to display triples for knowledge visualization. As shown in Figure 4, KnowledgeGraphViewer is an abstract class, which contains a show method.

**Figure 4: Design of the knowledge graph viewer framework.**

Recommendation Method Framework Design:

We have designed a dedicated framework as shown in Figure 5. DataPreprocessor module includes three methods, which are preprocessKG, preprocessUserInfo and preprocessRating. preprocessKG is used to process knowledge graph triples. It returns three dictionaries to store the id corresponding to the product, the id corresponding to the relationship and the id corresponding to the character. preprocessUserInfo is used to encode user information, including the user's gender, age, and occupation. preprocessRating uses the three dictionaries stored in the previous step to convert the product and user ID in the rating information. It consists of four frozen spots, DataLoader, DataPreprocessor, MLModelBuilder and Predictor. DataLoader reads all the triples from the file generated in the previous step, and returns three lists, number of users, number of items, and number of relations. loadKG is used to load the KG file processed in the previous step, calculate the number of entities, the number of relations, and return these values. These parameters are used to create the data needed for building the prediction model. loadUsers loads the user information file processed in the previous step, calculates the number of users, genders, ages and jobs. These parameters are used to create a user information matrix when training the neural network. loadRatings loads the rating file, then calculates the number of items, and then divides the data into the training data, evaluation data and test data according to the ratio of 6:2:2. The model is built through MLModelBuilder.

After training, the model is used to predict the user's rating for a specific product. Predictor facilitates this prediction. It includes two methods, getUserInfo and predictScore. The user's ID is used to query the user's personal information. Three lists are returned, user's gender, age, and job information. predictScore returns a float value, which represents the user's rating for the product.

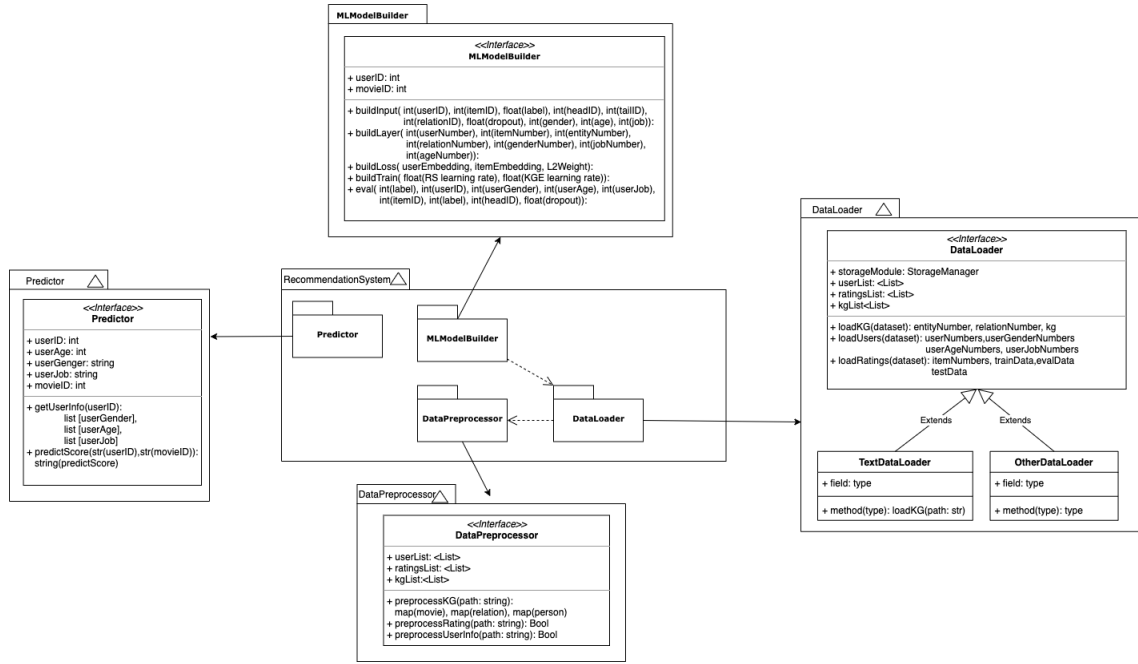


Figure 5: Design of the recommendation method framework.

4 FRAMEWORK INSTANTIATION

Instantiation is the process of specializing the framework to result in an executable recommendation system, for specific application environments. This is done by providing executable modules within frozen spots. Every frozen spot is specialized, depending on the need. Below we discuss a few illustrative examples.

IMDBExtractor Instantiation: Earlier, we described the design of the InfoExtractor module within the framework. If instantiated for movie recommendation, we extract director, writer, stars information, movie genre, movie poster and other available movie data as side information. In IMDBExtractor module, we create a list for each kind of information to be extracted. Because there may be many directors, actors, and stars of the movie, the information for each category is returned as a list. If the relevant information cannot be found in IMDB, an empty list will be returned. This is the incomplete information case. Each triplet will be stored in the form of head, relation, tail.

StorageManager Instantiation: We designed the frozen spot for a knowledge graph StorageManager framework in general. We created two storage modules as hot spots for knowledge graphs. They are Neo4jManager and RDFManager to accommodate the two different knowledge graph formats. Figure 6 shows the structure of this module. Figure 7 illustrates the structure of Neo4j storage. Use of the RDFManager requires operations of owl, including: RDFSave, RDFGetOntology, RDFAddClass, RDFAddIndividual, RDFAddDataproperty, RDFAddDatapropertyValue and RDFAddObjectproperty. Figure 8 illustrates the structure of RDF storage. When the user chooses to use RDF storage, it will call the RDFManager in the framework, use the API in our framework to operate on the triples, and then save it as an RDF file.

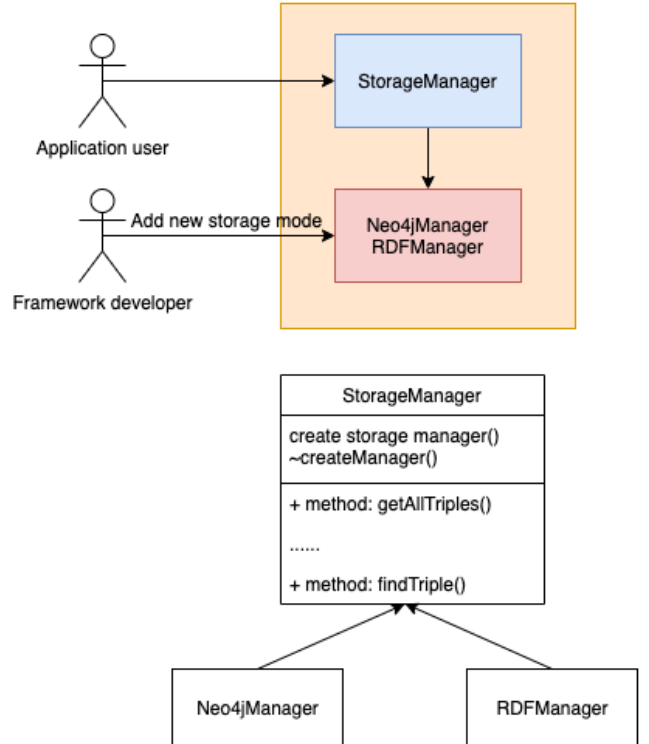


Figure 6: Instantiation of the StorageManager in framework.

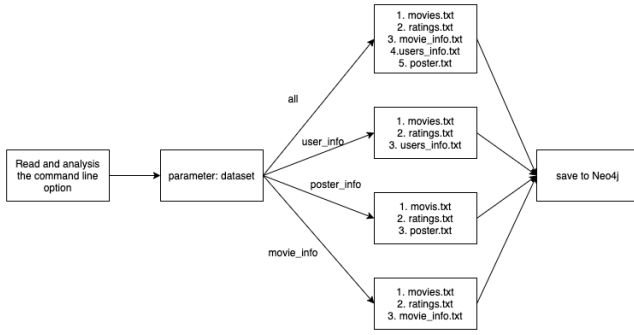


Figure 7: Structure of the Neo4j storage module in framework.

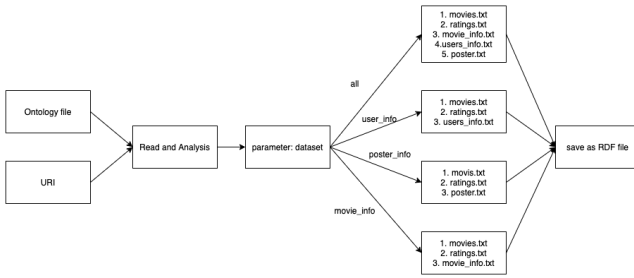


Figure 8: Structure of the RDF storage module in framework.

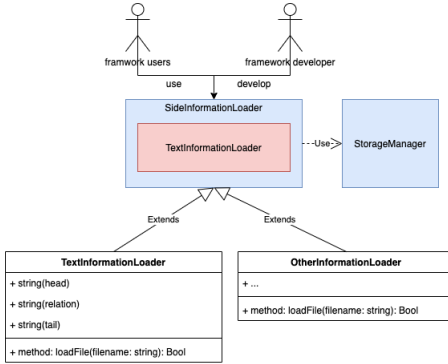


Figure 9: TextInformationLoader instantiation.

TextInfomationLoader: To add additional (side) information when available, based on the frozen spot SideInformationLoader, we created the TextInformationLoader hot spot. The overall idea is shown in Figure 9. We read the file in the parameter, parse the file according to the format, and extract the head, relation and tail of the triples. Then add new triples to the knowledge graph through addTriple in StorageManager. Since there is already a method for adding a single file, we only need to make some adjustments to loadFiles. When reading multiple files, we just need to call loadFile for each file.

networkxViewer Instantiation: The workflow of the viewer module is shown in Figure 10. Our process can be described by the

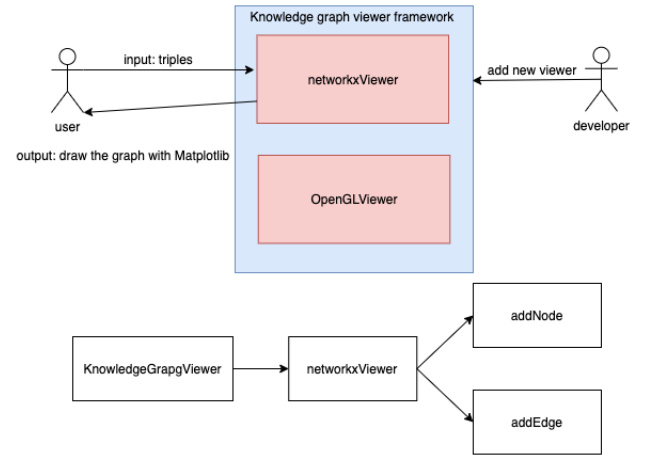


Figure 10: Instantiation of the networkxViewer framework.

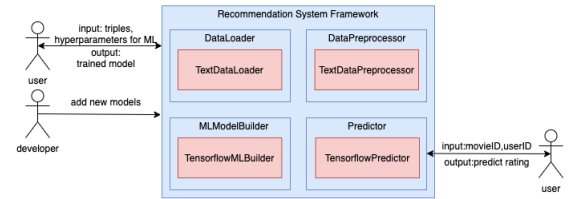


Figure 11: Workflow of recommender system framework

following steps:

- (1) Read all the individual names and store all the individuals in viewerIndividuals.
- (2) Take the individual list from the previous step and get all the information connected to this individual.
- (3) According to the content in the individual, create nodes or links respectively.

Recommendation Method Instantiation: The recommendation method is the most important part of our framework. Here we present a specialization of the recommendation method frozen spot, using a deep learning method. We decided to implement the entire method ourselves, as we wanted to incorporate the benefits of side information obtained from using knowledge graphs. While it is based on the work of [11, 13], our recommendation method[19] is written in Python and built on top of TensorFlow. Figure 11 illustrates the workflow of the recommendation method module. We suitably modified and implemented the network architecture of the

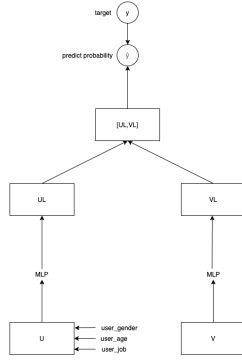


Figure 12: Structure of the recommendation module in RS framework.

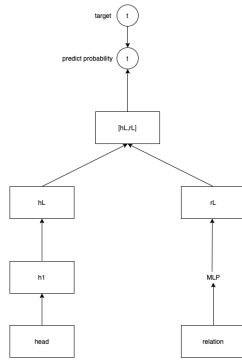


Figure 13: Structure of the knowledge graph embedding module in recommender system framework.

MKR method. The structure is shown in Figure 12. The original work does not contain user demographics, such as `user_gender`, `user_age` and `user_job` embeddings, which we have added for training. To unify the latitude (in the deep learning network), we also choose `arg.dim` as the dimension of `user_age`, `user_job` and `user_gender`.

For the knowledge graph embedding model, the structure is shown in Figure 13. Let's take our MovieLens data as an example. There are 3883 heads and four relations in the data. Therefore, the item matrix is the $3883 \times \text{arg.dim}$ matrix and the relation matrix is the $4 \times \text{arg.dim}$ matrix. Each time, we take out the vector corresponding to the head from the head embedding according to the head index, and then process the crossover and compression unit to obtain an $H_I b$ (head) vector. The R_I (relation) vector is obtained by looking up the vector of the relation index in the relation matrix and then passing through a fully connected layer. Then we merge the $[batch_size, \text{dim}]$ dimension H_I and R_I into a $[batch_size, 2 \times \text{arg.dim}]$ vector, and this vector is passed through a fully connected layer to get a $[batch_size, \text{arg.dim}]$ vector. This vector includes the predicted tail. For the cross and compress unit, item and head are each a vector of dimension $[batch_size, \text{arg.dim}]$. To facilitate calculation, we first expand them by one dimension so that they become $[batch_size, \text{arg.dim}, 1]$ and $[batch_size, 1, \text{arg.dim}]$ respectively, and then multiply them, to get the cross matrix c_matrix , a matrix of

$[batch_size, \text{dim}, \text{dim}]$, and a transpose matrix $c_matrix_transpose$, the two matrices are multiplied by different weights, then reshaped to obtain the final vectors.

During the training, our model is guided by three loss functions: L_{RS} loss and L_{KGE} loss and L_{REG} loss.

$$L = L_{RS} + L_{KG} + L_{REG} \quad (1)$$

The complete loss function is as follows:

- The first item is the loss in the recommendation module.
- The second item is the loss of the KGE module, which aims to increase the score of the correct triplet and reduce the score of the wrong triplet.
- The third item is L2 regularization to prevent overfitting.

Predictor: Once the model is trained, we can make predictions using our predictor module. There are two methods in this module, `getUserInfo` and `predictScore`. `getUserInfo` is a method used to extract user's information. The model trained in the previous step is loaded into `predictScore`, and then different matrices are read by name. If the id entered by the user is greater than the dimension of the model. That means the id is a new user. Our recommendation will focus on the user's age and job. Finally, a float value (predicted rating) is returned. These steps are described in Algorithm 1.

Algorithm 1: prediction application procedure

```

1 parser ← init a argument parser
2 dataset ← init default dataset
3 userid ← init default userid
4 movieid ← init default movieid
5
6 if trained model path exist then
7   load trained model
8   predict user's rating to movie
9 else
10  Error
  
```

Deployment: Figure 14 shows the diagram of all the required libraries. the purpose of each is briefly described below:

- `requests` is used to issue standard HTTP requests in Python. It abstracts the complexity behind the request so that users can focus on interacting with the service and using data in the application.
- `bs4` library is a functional library responsible for parsing, traversing, and maintaining the HTML "tag tree".
- `IMDB` is an online database of movie information.
- `Base64` is a library that uses 64 characters to represent arbitrary binary data.
- `py2neo` can use `Neo4j` from within the Python application and from the command line.
- `owlready2` is a module for ontology-oriented programming in Python.
- `rdflib` is used to parse and serialize files in RDF, owl, JSON and other formats.
- `networkx` is a graph theory and complex network modelling tool developed in Python language, which can facilitate complex network data analysis and simulation modelling.

- matplotlib is a data visualization tool.
- linecache and tensorflow libraries.
- random library is used to generate random numbers.
- NumPy is a math library mainly used for array calculations.
- sklearn is an open-source Python machine learning library that provides a large number of tools for data mining and analysis.
- linecache is used to read arbitrary lines from a file.
- TensorFlow is a powerful open-source software library developed by the Google Brain team for deep neural networks.

5 FRAMEWORK APPLICATION

Integrated Lenskit Application: This Lenskit application is a comparison of NDCG (Normalized Discounted Cumulative Gain) values for Lenskit recommendation algorithms. Figure 15 shows the result of the evaluation.

Prediction of User’s Rating for a Movie: So far, what we have described in the previous sections are the design, implementation and instantiation of our framework. Our prediction service predicts users’ ratings of products (as an example product we have chosen movies). The workflow can be described as follows:

- (1) Read all the triple data through dataLoader to get the corresponding information.
- (2) Use these triples and the user’s rating of the movie as input, and train through the RS module.
- (3) Load the trained model, and predict the user’s rating of the movie.

For the prediction task, we use the pipeline shown in Figure 16. Now, with this pipeline instance, application developers no longer need to worry about details. All the user needs to do is provide the right input.

Predict User’s Rating for a Book: The data we use is called Book-Crossing [5]. Book-Crossing dataset is Collected by Cai Nicolas Ziegler from the Book-Crossing community. The Book-Crossing dataset comprises 3 tables. They are users, Books and Ratings. Users contain the user’s id, the user’s location, and the user’s age. Books include book title, book author, year of publication, publisher and other information. Ratings include user reviews of the book. Because the process used is the same as for movies, we won’t repeat it here.

Knowledge Graph Fusion Application: Knowledge fusion is an effective way to avoid node duplication in knowledge graphs. Users may have knowledge graph files in different formats which may result in duplicate nodes with the same information. We use knowledge graph fusion to solve the problem of duplicate nodes. In our implementation, we provide an API where users can convert Neo4j triples to RDF or convert them to Neo4j based on RDF triples according to their needs. Algorithm 2 describes this procedure.

6 FRAMEWORK EVALUATION

Evaluation Testbed Specifications: Before starting the discussion about the evaluation, we first describe the environment, including the operating system used, processor power, memory, hardware, etc. The detailed specifications can be seen in Table 2. Table 3 lists the various libraries used in this implementation.

Algorithm 2: Knowledge graph fusion application procedure

```

1 parser ← init a argument parser
2 mode ← init default storage mode
3 path ← init default path
4
5 if args.mode == "neo4j2RDF" then
6   call Neo4j to RDF API
7 else
8   call RDF to Neo4j API
9
10 Neo4j_triples_list ← init a list for all the triples in Neo4j
11 RDF_triples_list ← init a list for all the triples in RDF
12 while read file do
13   add triples to Neo4j_triples_list
14   add triples to RDF_triples_list
15   while read Neo4j_triples_list or RDF_triples_list do
16     if node exist in RDF_triples_list or Neo4j_triples_list
17       then
18         find the corresponding node
19       else
20         create new node
21   connect nodes and relations

```

Setting	Name	Device
Laptop	Memory	8 GB
	Processor	2.3 GHz Intel Core i5
	Graphic	Intel Iris Plus Graphics 640 1536 MB
	OS	Mac OS Mojave 10.14.6
Server (colab)	Memory	12 GB
	Processor	Intel Core i7-920 CPU 2.67GHz
	Graphic	GeForce GTX1080 Ti (12 GB)
	OS	Ubuntu 18.04.5 LTS 64-bit

Table 2: Environment hardware specifications.

Type	Name	Version
Libraries	py2neo	4.3.0
	Tensorflow	1.14.0
	bs4	4.8.0
	csv	1.0
	networkx	2.4
	matplotlib	3.1.2
	rdflib	4.2.2
	numpy	1.17.0
	sklearn	0.21.3
	pandas	0.25.0

Table 3: Python libraries used.

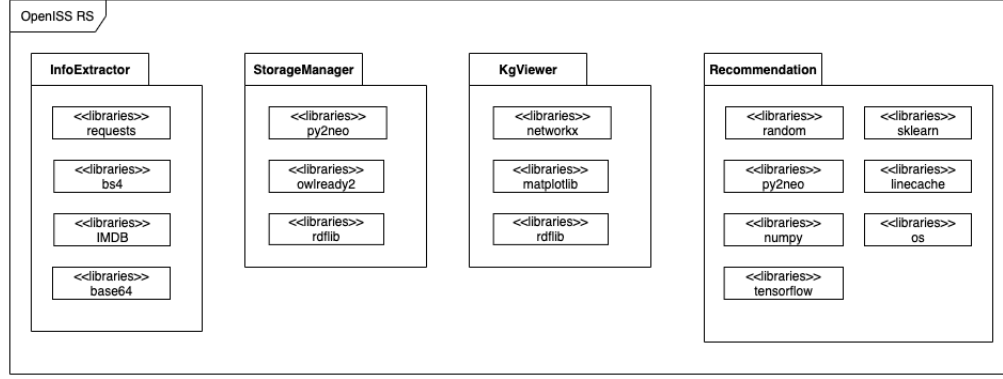


Figure 14: UML deployment diagram.

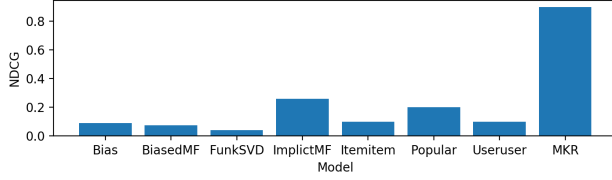


Figure 15: LensKit evaluation result.

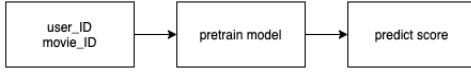


Figure 16: Prediction pipeline.

Due to different operating systems, the software is slightly different, so we also need to give the software details of the environment. For these two environments: one is a laptop and the other is a server, we call them Setting 1 and Setting 2, respectively. There is a slight difference between the installed software versions. For these two environments, we give more detailed information in Table 4.

Software	Setting 1	Setting 2
Python	3.7.5	3.7.5
Neo4j Desktop	1.2.1	1.2.9
Protege	5.5.0	5.5.0

Table 4: Software packages and IDE tools used.

Real-time Response: According to [20], we set the real-time response baseline to be 2000 ms. To evaluate the whole system’s processing ability, we performed the experiment described below:

- (1) Load the trained model to get the pre-trained graph structure and weights.
- (2) Prepare feed_dict, new training data or test data. In this way, the same model can be used to train or test different data..
- (3) Measure the difference between the timestamp t_s before the pipeline start and the timestamp t_d after system processing.

- (4) Repeat the previous operation 100 times, and then calculate the average processing time through the formula Equation 2.

$$Result = \frac{\sum_{i=1}^{100} t_e - t_s}{100}. \quad (2)$$

The result shows that the speed of our solution on a local laptop machine is 634.33ms. It is faster than the real-time baseline of 2000ms.

Experimental Results: We trained the MKR model using the MovieLens-1m dataset, and then used the validation set to verify the model. We split all data according to 6:2:2, i.e., 60% is the training set, 20% is the validation set, and 20% is the test set. The data of the validation set and test set were not be used for training. Our side information can be of many types, including movie information, user information, and movie posters. We train with different side information through our model and obtain the results through 20 epoch training, as shown in Table 5.

From the results, we can see that the accuracy of data with user information and movie information is the highest, about 1% higher than the baseline. Because users may watch a movie because of a director or an actor, the other movie information can help improve accuracy. The age, job and other information in the user information also help to improve the accuracy, because the user may choose some related movies to watch based on age and occupation. But because the poster of each movie is different, in the knowledge graph, each poster is connected to only one movie node, so the poster data is sparse data for the knowledge graph. Therefore, the poster information does not have a good effect for us at present, but if we can extract some useful information from the poster through technologies such as computer vision, it may be helpful in improving the accuracy of the recommendation.

7 CONCLUDING REMARKS AND FUTURE EXTENSIONS

We proposed, designed and implemented a generic software framework that integrates knowledge graph representations for providing training data to enhance the performance of deep learning based recommendation methods. We demonstrate the improvements in performance for the domain of movie recommendations

MovieLens 1M	train AUC	train ACC	evaluate AUC	evaluate ACC	test AUC	test ACC
baseline	0.9189	0.8392	0.9046	0.8277	0.9042	0.8261
baseline+movie	0.9227	0.8439	0.9081	0.8295	0.9061	0.8297
baseline+user	0.9238	0.8455	0.9096	0.8321	0.9091	0.8331
baseline+user+movie	0.9292	0.8516	0.9142	0.8375	0.9136	0.8359
baseline+poster	0.9173	0.8279	0.9041	0.8153	0.9029	0.8214
baseline+movie+user+poster	0.9273	0.8497	0.9113	0.8351	0.9111	0.8349

Table 5: Results of the same models on different datasets.

to users made possible by the additional information that is extracted through knowledge graphs. At the core of this framework are knowledge graphs for storing and managing information for use by recommendation methods. The framework is generic and can be specialized to accommodate different product domains, recommendation algorithms, data gathering strategies, and knowledge graph storage formats. To the best of our knowledge, a similar framework is not available elsewhere.

The ultimate goal of our work is to make it as a research platform for more developers in the recommender systems field. With that goal our framework needs to be extended as follows:

Java API wrapper: Our framework was written in Python, but the movie recommender system is mostly used on web pages. So it is better for users, if can we provide a Java wrapper for our API.

Support different machine learning backends: Currently, our recommendation module only supports TensorFlow. But there are many different deep learning frameworks, such as PyTorch, Caffe or Scikit-learn. Different frameworks have their advantages. We plan to add various machine learning frameworks to our framework in the future.

Support more storage methods and more input formats: Currently, we only support four storage formats, namely RDF, RDFS, OWL and Neo4j. For the input format, because we use CSV for storage, some users may choose JSON format or TTL format, so we also need to update the program to support these formats.

REFERENCES

- [1] [n.d.]. The Amazon Recommendations Secret to Selling More Online. <http://rejoiner.com/resources/amazon-recommendations-secret-selling-online>. Accessed: 2018-03-27.
- [2] [n.d.]. Deep Dive into Netflix Recommender System. <https://towardsdatascience.com/deep-dive-into-netflixs-recommender-system-341806ae3b48>. Accessed: 2016-04-30.
- [3] [n.d.]. Recommender system — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/wiki/Recommender_system. Accessed: 2020-07-27.
- [4] [n.d.]. YouTube recommendations drive 70 percent of what we watch. <https://qz.com/1178125/youtubes-recommendations-drive-70-of-what-we-watch/>. Accessed: 2018-01-13.
- [5] 2004. Book-Crossing Dataset. [online], wikipedia. <http://www2.informatik.uni-freiburg.de/~cziegler/BX/>.
- [6] 2006–2011. LensKit recommendation framework. [online], wikipedia. <https://github.com/lenskit/lkpy>.
- [7] 2013. recommendation-GER. [online], wikipedia. <https://github.com/grahamjenson/ger>.
- [8] 2013. recommendation-Raccoon. [online], wikipedia. <https://github.com/guymorita/recommendationRaccoon#recommendationraccoon-raccoon>.
- [9] G. Adomavicius and A. Tuzhilin. 2005. Towards the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Transactions on Knowledge and Data Engineering* 17, 6 (June 2005), 734–749. <https://doi.org/10.1145/223904.223929>
- [10] Sören Auer, Viktor Kovtun, Manuel Prinz, Anna Kasprzik, Markus Stocker, and Maria Esther Vidal. 2018. Towards a knowledge graph for science. In *Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics*. 1–6.
- [11] Rianne van den Berg, Thomas N Kipf, and Max Welling. 2017. Graph convolutional matrix completion. *arXiv preprint arXiv:1706.02263* (2017).
- [12] Niel Chah. 2018. OK Google, What Is Your Ontology? Or: Exploring Freebase Classification to Understand Google’s Knowledge Graph. arXiv:1805.03885 [cs.IR]
- [13] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for YouTube Recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*. New York, NY, USA.
- [14] Xin Dong, Evgeniy Gabrilovich, Jeremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 601–610.
- [15] Michael D Ekstrand, John T Riedl, and Joseph A Konstan. 2011. *Collaborative filtering recommender systems*. Now Publishers Inc.
- [16] Qingyu Guo, Fuzhen Zhuang, Chuan Qin, Hengshu Zhu, Xing Xie, Hui Xiong, and Qing He. 2020. A Survey on Knowledge Graph-Based Recommender Systems. arXiv:2003.00911 [cs.IR]
- [17] Tom Hanika, Maximilian Marx, and Gerd Stumme. 2019. Discovering Implicational Knowledge in Wikidata. arXiv:1902.00916 [cs.AI]
- [18] Nicolas Heist, Sven Hertling, Daniel Ringler, and Heiko Paulheim. 2020. Knowledge Graphs on the Web – an Overview. arXiv:2003.00719 [cs.AI]
- [19] Yuhao Mao, Serguei A. Mokhov, and Sudhir P. Mudur. 2021. Application of Knowledge Graphs to Provide Side Information for Improved Recommendation Accuracy. *CoRR* abs/2101.03054 (2021). arXiv:2101.03054 <https://arxiv.org/abs/2101.03054>
- [20] Jan Mizgajski and Miko Morzy. 2019. Affective Recommender Systems in Online News Industry: How Emotions Influence Reading Choices. *User Modeling and User-Adapted Interaction* (April 2019), 35. <https://doi.org/10.1007/s11257-018-9213-x>
- [21] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. 2015. A review of relational machine learning for knowledge graphs. *Proc. IEEE* 104, 1 (2015), 11–33.
- [22] Yue Ning, Yue Shi, Liangjie Hong, Huzefa Rangwala, and Naren Ramakrishnan. 2017. A gradient-based adaptive learning framework for efficient personal recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. 23–31.
- [23] Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. DKN: Deep Knowledge-Aware Network for News Recommendation. arXiv:1801.08284 [stat.ML]
- [24] Hongwei Wang, Fuzheng Zhang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2019. Multi-Task Feature Learning for Knowledge Graph Enhanced Recommendation. arXiv:1901.08907 [cs.IR]
- [25] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering* 29, 12 (2017), 2724–2743.
- [26] Zhanfang Zhao, Sung-Kook Han, and In-Mi So. 2018. Architecture of knowledge graph construction techniques. *International Journal of Pure and Applied Mathematics* 118, 19 (2018), 1869–1883.

Sink Group Betweenness Centrality

Evangelia Fragkou
University of Thessaly
Volos, Greece
efragkou@e-ce.uth.gr

Dimitrios Katsaros
University of Thessaly
Volos, Greece
dkatsar@e-ce.uth.gr

Yannis Manolopoulos
Open University of Cyprus
Nicosia, Cyprus
yannis.manolopoulos@ouc.ac.cy

ABSTRACT

This article introduces the concept of Sink Group Node Betweenness centrality to identify those nodes in a network that can “monitor” the geodesic paths leading towards a set of subsets of nodes; it generalizes both the traditional node betweenness centrality and the sink betweenness centrality. We also provide extensions of the basic concept for node-weighted networks, and also describe the dual notion of Sink Group Edge Betweenness centrality. We exemplify the merits of these concepts and describe some areas where they can be applied.

CCS CONCEPTS

• **General and reference** → **Metrics; Evaluation;** • **Human-centered computing** → **Collaborative and social computing design and evaluation methods; Social network analysis; Social engineering (social sciences);**

KEYWORDS

Sink Group Betweenness Centrality, Sink Group Edge Betweenness Centrality, Betweenness Centrality, Network Science

ACM Reference Format:

Evangelia Fragkou, Dimitrios Katsaros, and Yannis Manolopoulos. 2021. Sink Group Betweenness Centrality. In *25th International Database Engineering & Applications Symposium (IDEAS 2021), July 14–16, 2021, Montreal, QC, Canada*. ACM, New York, NY, USA, 6 pages. <https://doi.org/10.1145/3472163.3472182>

1 INTRODUCTION

The dramatic growth of online social networks during the past fifteen years, the evolution of Internet of Things and of the emerging Internet of Battlefield Things, the extensive study and recording of large human social networks is offering an unprecedented amount of data concerning (mainly) binary relationships among ‘actors’. The analysis of such graph-based data becomes a challenge not only because of their sheer volume, but also of their complexity that presents particularities depending on the type of application that needs to mine these data and support decision making. So, the field termed *network science* has spawned research in a wide range of topics, for instance: a) in network growth, developing models

such as the preferential attachment [4], b) in new centrality measures for the identification of the most important actors in a social network developing measures such as the bridging betweenness centrality [28], c) in epidemics/diffusion processes [18], d) in finding community structure [21, 22], i.e., finding network compartments which comprise by sets of nodes with a high density of internal links, whereas links between compartments have comparatively lower density, e) in developing new types of networks different from static and single-layer networks, such temporal [40] and multilayer networks [9], and of course analogous concepts for these types of networks such as centralities [26], communities [27], epidemic processes [5], and so on.

Despite the rich research and the really large number of concepts developed during the past twenty years and the diverse areas where it has been applied e.g., ad hoc networking [30], the field of network is continuously flourishing; the particular needs of graph-data analytics create the need for diverse concepts. For instance, let us examine a very popular and well-understood concept in the analysis of complex networks which is the notion of centrality [3, 35, 42], being either graph-theoretic [23], or spectral [34] or control theoretic [29]. Betweenness centrality [23] in particular has been very successful and used for the design of effective attacks on network integrity, and also for discovering good “mediators” (nodes able to monitor communication among any pair of nodes in a network), but it is not effective in identifying influential spreaders; for that particular problem k -shell decomposition [31] proved a much better alternative. However subsequent research [11] proved that a node’s spreading capabilities in the context of rumor spreading do not depend on its k -shell index, whereas other concepts such as the PCI index [6] can perform better. So, our feeling is that the field of network science, even some very traditional concepts such as betweenness centrality could give birth to very useful and practical variants of them.

Let us describe a commonly addressed problem in network science concerning malware spreading minimization problem. In particular, we are given a set of subsets of computer nodes that we need to protect against a spreading malware which has already infected some nodes in the network, but we only have a limited number of vaccines (i.e., a “budget”) to use. If we had to select some healthy (susceptible) nodes to vaccinate, then which would these nodes be? Our decision would of course depend on the infection spreading model, but usually routing in computer networks implements a shortest-path algorithm. So can the traditional shortest-path node betweenness centrality [23] help us to identify such nodes?

Additionally, we will describe a similar problem that a modern army could possibly face due to the rapid deployment of Internet of Battlefield Things [44]. The army needs to destroy by jamming the communications towards a set of subset of nodes of the enemy, but

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](https://permissions.acm.org).
IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada
© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-8991-4/21/07...\$15.00
<https://doi.org/10.1145/3472163.3472182>

it has available only a limited number of jammer or time to deploy them. The question is now which links should the army select to attack? So can the traditional shortest-path edge betweenness centrality [15] help us to identify such links?

The answer to previous questions is negative, because node/edge betweenness centrality calculates the importance of a node/edge lying on many shortest paths connecting any pair of network nodes, whereas in our problems we are interested in paths leading towards specific sets of subsets of network nodes. Starting from this observation, in this article we introduce the generic concept of *sink-group betweenness centrality* which can be used as a building block for designing algorithms to address the aforementioned problems.

The aim of the present article is to introduce the new centrality concept for various types of complex networks and also to present some potential uses of it for solving some network science problems. In this context, the present article makes the following contributions:

- it introduces a new centrality measure, namely the sink group node betweenness centrality;
- it extends the basic definition for networks weighted on the nodes;
- it extends the basic definition for the case of edge betweenness;
- it presents basic algorithmic ideas for calculating the aforementioned notions.

The rest of the article is structured as follows: section 2 presents the articles which are closely related to the present work; section 3 defines the sink group betweenness centrality concept, and in section 4 we provide a detailed example to exemplify the strengths of the new concept. Then, in sections 5 and 6 we present the definitions of sink-group betweenness centrality for node-weighted networks and edge sink group betweenness centrality, respectively. Finally, section 8 concludes the article.

2 RELATED WORK

Betweenness Centrality.

The initial concept of (shortest path) betweenness centrality [23] gave birth to concepts such as the proximal betweenness, bounded-distance betweenness and distance-scaled betweenness [13], bridging centrality [28] to help discover bridging nodes, routing betweenness centrality [16] to account for the paths followed by routed packets in a networks, percolation centrality [39] to help measure the importance of nodes in terms of aiding the percolation through the network. There are so many offsprings of the initial concept, that even a detailed survey would found practically impossible to record each one published!

The realization that the computation of betweenness centrality requires global topology knowledge and network-wide manipulation, which is computationally very expensive, spawned research into distributed algorithms for its computation [10], and inspired variants aiming at facilitating the distributed computation of notions similar to the original betweenness centrality, such as load centrality [37].

Betweenness centrality and its variations have found many applications not only in classical fields in network mining, but also

in delay-tolerant networks [38], ad hoc networks [30], in distributed systems e.g., for optimal service placement [43], etc.

Approximate Betweenness Centrality.

Many applications working over modern networks require the calculation of betweenness centrality in nearly a real-time fashion or have to deal with a huge number of nodes and edges. So, the decision of trading off the accuracy of betweenness centrality computation with speed arises as a natural option. Some early works considered approximating the exact values of betweenness centrality [2, 24]. Later on, this became a very active research field [10]; a survey can be found at [41].

Group Betweenness Centrality.

Group betweenness centrality indices [19] measure the importance of groups of nodes in networks, i.e., they measure the percentage of shortest paths that pass through at least one of the nodes of the group. On the other hand, co-betweenness centrality [32] measures the percentage of shortest paths that pass through all vertices of the group.

Algorithms for fast calculation of these group centrality measures have been developed [14, 45] even for diverse types of networks [36]; group (co-)betweenness or their variations have found applications in monitoring [17], in network formation [8], etc.

Sink Betweenness Centrality.

The notion of Sink Betweenness centrality [46], which is a specialization of our Sink Group Betweenness Centrality, was developed in the context of wireless sensor network to capture the position of nodes which lie in many paths leading to a specific node, i.e., the sink. So, the sink betweenness of a sensor node was correlated to the energy consumption of than node, since it had to relay a lot of messages towards the sink node.

3 THE SINK GROUP BETWEENNESS CENTRALITY

We assume a complex network $G(V, E)$ consisting of n nodes, where $V = \{v_i, 1 \leq i \leq n\}$ is the set of nodes and $E = \{(v_i, v_j), i, j \in V\}$ is the set of edges. We make no particular assumptions about the network being directed or undirected; this will be handled seamlessly by the underlying shortest-path finding algorithm. We assume that the network is unweighted, that is, neither the nodes nor the edges carry any weights; however, the former assumption will be reconsidered in section 5.

Recall that the goal of sink group betweenness centrality is to discover which nodes lie in many paths leading towards a particular set of designated nodes. To achieve our purpose we combine the concepts of Group Betweenness (\mathcal{GBC}) [20, 33] (although in a different fashion than in the initial definition) and the concept of Sink Betweenness (\mathcal{SBC}) [46]. In the following we will define the measure of Sink Group Node Betweenness Centrality (\mathcal{SGBC}), but firstly we will remind to the reader some definitions.

Recall that the Shortest Path Betweenness (\mathcal{SPBC}) Centrality is defined as follows¹:

Definition 3.1 ((Shortest Path) Betweenness Centrality [23]). The (Shortest Path) Betweenness Centrality (\mathcal{SPBC}) of a node v is the

¹When using the term “betweenness centrality” we refer to the concept of node betweenness centrality. When we wish to refer to the “edge betweenness centrality” we will explicitly make use of this term.

fraction of shortest paths between any pair of nodes that v lies on. Equation 1 calculates the $SPBC$ of node v .

$$SPBC(v) = \sum_{i=1}^n \sum_{\substack{j=1 \\ i \neq j \neq v}}^n \frac{\sigma_{ij}(v)}{\sigma_{ij}} \quad (1)$$

where σ_{ij} is the number of shortest paths from node i to j , and $\sigma_{ij}(v)$ is the number of shortest paths from i to j where node v lies on. This is the non-normalized version of betweenness centrality, i.e., we do not divide by the total number of node pairs.

Definition 3.2 (Sink Betweenness Centrality [46]). The Sink Betweenness Centrality (SBC) of a node v is the fraction of shortest paths leading to a *specific* sink node s , that v lies on. Formally, we provide Equation 2 for calculating the SBC of node v .

$$SBC(v) = \sum_{\substack{i=1 \\ i \neq s \neq v}}^n \frac{\sigma_{is}(v)}{\sigma_{is}} \quad (2)$$

s is the sink node

Apparently, SBC is a specialization of $SPBC$ i.e., j does not iterate over all nodes of the network but it is kept fixed and coincides with the sink node s ($j \equiv s$).

Suppose now that there is some “abstract grouping” process that defines non-overlapping clusters of nodes over this network. In principle, we do not need to apply any grouping algorithm at all, but we can assume that some application selects the members of each cluster as part of our input. The union of these clusters may not comprise the whole complex network. We expect that Usually the size of the aggregation of all these clusters comprises a small fraction of the complex network. Let us assume that we have defined z non-overlapping clusters, namely C_1, C_2, \dots, C_z with cardinalities $|C_1|, |C_2|, \dots, |C_z|$, respectively. Then, the *Sink Group Betweenness Centrality* is defined as follows:

Definition 3.3 (Sink Group Betweenness Centrality). The Sink Group Betweenness Centrality ($SGBC$) of a node v is the fraction of shortest paths leading to any node, which is a member of any designated cluster, that v lies on. Formally, we provide Equation 3 for calculating the $SGBC$ of node v .

$$SGBC(v) = \sum_{i=1}^n \sum_{\substack{j \in \bigcup_{k=1}^z C_k \\ v \notin \bigcup_{k=1}^z C_k}} \frac{\sigma_{ij}(v)}{\sigma_{ij}} \quad (3)$$

where n is the number of nodes, C_k is the k -th cluster, σ_{ij} is the number of shortest paths from node i to j , and $\sigma_{ij}(v)$ is the number of shortest paths from i to j where node v lies on.

Intuitively, a node has high Sink Group Betweenness Centrality if it sits in many shortest paths leading towards nodes belonging to any group.

Definition 3.3 requires that the node whose $SGBC$ we calculate is not part of any existing cluster. This is not mandatory in general, but since we are looking for nodes which can act as mediators in the communication with the clusters’ nodes, it makes sense to exclude the clusters’ nodes from being considered as potential mediators. By removing this constraint, we get the concept of *Generalized SGBC*, but in this article we consider it to be equivalent to the plain $SGBC$.

We have considered unweighted and undirected complex networks. The extension of the definition of Sink Group Betweenness Centrality for directed networks is straightforward, because it is handled by the path finding algorithm. On the other hand, the extension to node or edge weighted networks is less straightforward and we will provide some ideas in a later section.

3.1 $SGBC$ versus its closest relatives

$SGBC$ has as its special cases the concepts of betweenness centrality and of sink betweenness centrality. In particular, $SGBC$ is related to its closest relatives as follows:

- Clearly, $SGBC$ is related to the $SPBC$ in the following way: when the union of all clusters comprises the whole network, then *Generalized SGBC* and $SPBC$ coincide.
- Moreover, $SGBC$ is a generalization of SBC in the following way: when we have only one cluster which contains a single node, then $SGBC$ and SBC coincide.
- At this point we need to make clear the difference between Group Betweenness Centrality [20] and $SGBC$; the former seeks for the centrality of a group of nodes with respect to the rest of the nodes of the network, whereas the latter seeks for the centrality of a single node with respect – not the nodes of the whole network but – to the nodes belonging to some groups (clusters). Apparently, we can generalize Definition 3.3 and Equation 3 to follow the ideas of [20].

4 EXEMPLIFYING THE MERIT OF $SGBC$

Let us now look at the small complex network of Figure 1. This network represents a collection of communicating nodes administered by an overseeing authority. We have no weights on links, but we have denoted the nodes that are mostly important for the authority – and thus must be protected better – with red color. We have not designated any attacker or attacked node in the figure, because in many situations the attack might be known where it will initiate.

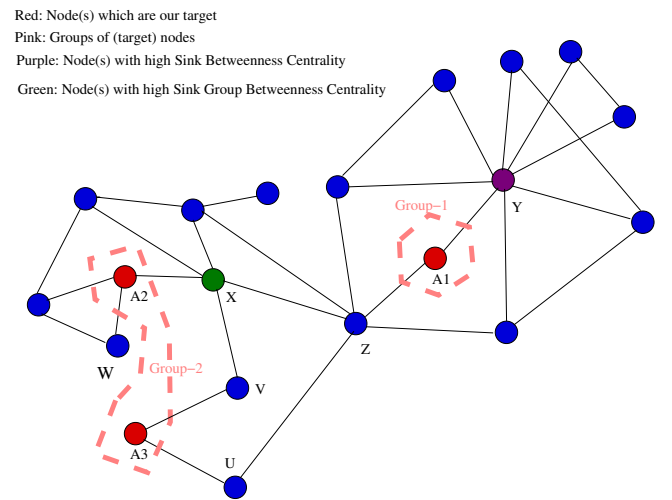


Figure 1: Illustration of Sink Group Betweenness Centrality.

Suppose that this authority is interested in *investing a limited amount of money to buy hardware/software to equip some nodes with* hoping that these will protect the significant ones, e.g., by stopping any cascade of infections. Let us call such nodes *safeguarding nodes*. Moreover, it is obvious that the authority needs to identify a limited number of safeguarding nodes, due to the limited budget. So, how we identify safeguarding nodes by exploiting the topological characteristics of the complex network?

The obvious solution is to look at the one-hop neighbors of the safeguarding nodes. Then, we can make use of the *SBC* [46] and say that the purple node (*Y*) is a safeguarding node. The purple node sits on many shortest paths towards the red node *A1* (Group-1). In this way, we can select the set of nodes $\{V, W, Y\}$ as safeguarding nodes.

When the constraint of reducing the cardinality of this set comes into play, then we must somehow group safeguarding nodes, and seek for safeguarding nodes which are close to each group or to many groups. (This tradeoff will be analysed in the sequel.)

Concerning the structure of groups, we have the following characteristics:

- Clusters might contain only one node.
- Nodes comprising a cluster need not be one-hop neighbors.

SBC is of no use anymore, but we can use the concept of *SGBC* defined earlier. Using this concept, the green node *X* has high Sink Group Betweenness Centrality because it sits on many shortest paths towards the two red nodes *A2* and *A3* comprising Group-2. Notice, here that the nodes with high *SGBC* are not correlated to nodes with high *SPBC*; for instance, the node with the highest *SPBC* in the network is node *Z* (it is an articulation point or bridge node).

Now let us look at the impact of safeguarding nodes' grouping on the existence (and/or *SGBC* value) of safeguarding nodes. As said earlier, the grouping creates the following tradeoff: the larger the groups we define, the less the nodes (if any) with high Sink Group Betweenness Centrality we can find.

If we unite Group-1 and Group-2 into a single group and ask the question 'which node(s) sit in many shortest paths towards ALL members of this new group', then we can safely respond that only node *Z* is such a node, because of the particular structure of the network of Figure 1. Recall that node *Z* is a bridge node, thus all shortest paths from the right of *Z* to nodes *A2* and *A3* will pass via *Z*, and all shortest paths from the left of *Z* towards node *A1* will pass via *Z*.

If we now examine Figure 2, and consider initially a grouping comprised by two groups, i.e., Group-1 and Group-2, then we can clearly identify the two green nodes as those with the highest *SGBC*. If we create a single large group comprised by all red nodes, then we can not find any node with high enough *SGBC* to act as safeguarding nodes. For this grouping, the blue nodes have also *SGBC* comparable to that of green nodes. Thus, with this grouping the identification of safeguarding nodes becomes problematic.

4.1 Calculation of *SGBC*

The calculation of *SGBC* can be carried out using as basis the Dijkstra's algorithm. Algorithm 1 is a baseline one:

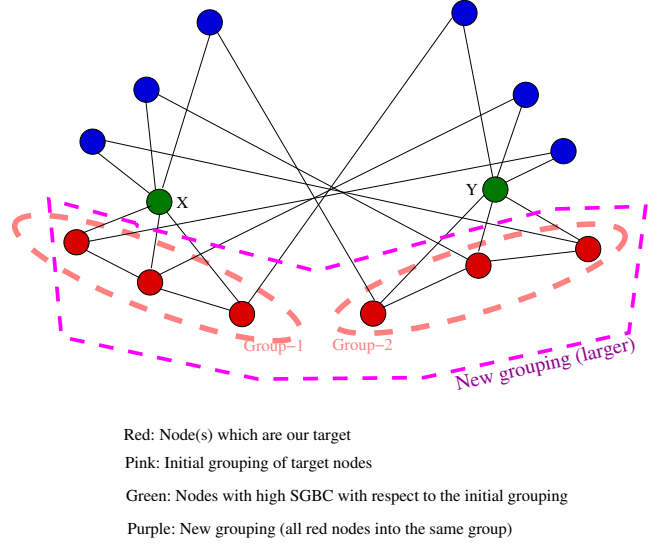


Figure 2: Impact of grouping on Sink Group Betweenness Centrality.

Algorithm 1: Calculation of *SGBC*.

```

1 for each node  $s \in \cup_{k=1}^z C_k$  to  $j \in \{V - \cup_{k=1}^z C_k\}$  do
2    $SP_s = \text{Dijkstra to find shortest-path from } s \rightarrow j;$ 
3  $SP = \cup_{s \in S} SP_s;$ 
4 for each path  $p \in SP$  do
5   Use a hash table to group paths based on start-end;
6 for each hash table bucket do
7   Use a hash table to accumulate node appearance in
   paths;
```

PROPOSITION 4.1. *The worst-case computational complexity of Algorithm 1 is $O(|\cup_{k=1}^z C_k| \times n^2)$, where n is the number of network nodes.*

PROOF. Assuming a network with n nodes and m edges, and an implementation² of Dijkstra's algorithm that costs $O(n^2 + m)$, i.e., $O(n^2)$, then Lines 1–2 cost $O(|\cup_{k=1}^z C_k| \times n^2)$; lines 3–5 cost $O(n^2)$ since there are at most n^2 paths, and lines 6–7 cost $O(n^2)$. \square

For unweighted and undirected networks, we can design a faster algorithm along the ideas of breadth-first traversal and the algorithm by Brandes [12], but this is beyond the scope of the present article.

5 *SGBC* WITH NODE WEIGHTS

Now suppose that each node has a weight associated with it (e.g., depicting its trustworthiness, its balance in a transaction network, etc), then we need to define the *SGBC* in such a way that it will take into account these weights. Note that this is different from having weights in edges (i.e., a weighted network) because that

²There are various implementations with even smaller cost utilizing sophisticated data structures or taking advantage of network sparsity.

weights are handled by the shortest-path finding algorithm. We can have the following options:

- We can apply the straightforward idea of multiplying by a node's weight as in [1], i.e.,

$$SGBC^{nw}(v) = node_weight(v) \times \sum_{i=1}^n \sum_{\substack{j \in \bigcup_{k=1}^z C_k \\ v \notin \bigcup_{k=1}^z C_k}} \frac{\sigma_{ij}(v)}{\sigma_{ij}} \quad (4)$$

with appropriate normalization in the interval $[0 \dots 1]$ for both node's weight and $SGBC$.

- Apply standard procedures for turning the problem of finding shortest paths in networks with weights on edges and/or nodes into a shortest path finding problems with weights only on edges. The methods are the following (n - number of nodes, m - number of edges):
 - We can split each node apart into two nodes as follows: for any node u , make two new nodes, u_1 and u_2 . All edges that previously entered node u now enter node u_1 , and all edges that leave node u now leave u_2 . Then, put an edge between u_1 and u_2 whose cost is the cost node u_1 . In this new graph, the cost of a path from one node to another corresponds to the cost of the original path in the original graph, since all edge weights are still paid and all node weights are now paid using the newly-inserted edges. Constructing this graph can be in time $O(m + n)$, since we need to change each edge exactly once and each node exactly once. From there, we can just use a normal Dijkstra's algorithm to solve the problem in time $O(m + n \log n)$, giving an overall complexity of $O(m + n \log n)$. If negative weights exist, then we can use the Bellman-Ford algorithm instead, giving a total complexity of $O(mn)$.
 - Alternatively, we can think as follows: since we have both edges and nodes weighted, when we move from i to j , we know that total weight to move from i to j is weight of edge($i \rightarrow j$) plus weight of j itself, so lets make $i \rightarrow j$ edge weight sum of these two weights and the weight of j zero. Then we can find shortest path from any node to any other node to in $O(m \log n)$.
- We can multiply each fraction (of shortest paths) in the summation formula with the minimum weight (positive or negative) found along the path.

6 SINK GROUP EDGE BETWEENNESS CENTRALITY

The plain edge-betweenness centrality measure is used to identify the edges which lie in many shortest-paths among pair of network nodes. It has been widely used in network science and not only, e.g., for discovering communities in networks [25], for designing topology control algorithms for ad hoc networks [15], etc.

In our context, we ask the question whether we can identify the edges which lie in many paths towards a set of subsets of network nodes. The extension of $SGBC$ to the case of edges is easy. Thus, the *Sink Group Edge Betweenness Centrality* is defined as follows:

Definition 6.1 (*Sink Group Edge Betweenness Centrality*). The Sink Group Edge Betweenness Centrality ($SGEBC$) of an edge

e is the fraction of shortest paths leading to any node, which is a member of any designated cluster, that e lies on. Formally, we provide Equation 5 for calculating the $SGEBC$ of edge e .

$$SGEBC(e) = \sum_{i=1}^n \sum_{\substack{j \in \bigcup_{k=1}^z C_k \\ v \notin \bigcup_{k=1}^z C_k}} \frac{\sigma_{ij}(e)}{\sigma_{ij}} \quad (5)$$

where n is the number of nodes, C_k is the k -th cluster, σ_{ij} is the number of shortest paths from node i to j , and $\sigma_{ij}(e)$ is the number of shortest paths from i to j where edge e lies on.

7 APPLICATIONS OF $SGBC$

7.1 $SGBC$ and influence minimization

We have already explained in the introduction how sink group betweenness centrality can be used to limit the spreading in the context of influence or infection minimization problems under budget constraints. Especially relevant becomes for those problems that are online [7], i.e., require a continuous combat against the spreading while the infection evolves in various parts of the network.

7.2 $SGBC$ and virtual currencies networks

Let us now look again at the network of Figure 1, but this time assume that the graph represents a network of transactions being made using a virtual currency, e.g., a community currency [47], which – differently from BitCoin – is administered by an overseeing authority. We have denoted the nodes in deficit (i.e., with lack of virtual money) with red color.

Suppose that this authority is interested in injecting a limited amount of money into some nodes with the hope that these nodes will buy something from the red nodes and therefore with reduce their deficit. Let us call such nodes *deficit balancers*. Suppose that the authority needs to identify a limited number of deficit balancers, otherwise will have to divide this amount of money into too many deficit balancers, and eventually an even smaller amount of money will end up to the nodes with deficit. So, how we identify deficit balancers?

Exactly as before, the obvious solution is to look at the one-hop neighbors of the deficated nodes. Then, we can make use of the SBC and say that the purple node (Y) is a deficit balancer. The purple node sits on many shortest paths towards the red node $A1$ (Group-1). In this way, we can select the set of nodes $\{V, W, Y\}$ as deficit balancers. If the authority is constrained to reduce the cardinality of this set, then we must seek for deficit balancers which are close to each group or to many groups. From this point, we can continue along the same reasoning as we did in section 4.

7.3 $SGBC$ and community finding

The concept of $SGEBC$ can be used as a component of an algorithm for discovering the community where a particular set of nodes belongs. The idea is that if this collection of nodes is relative close to each other, then repeated deletion of high $SGEBC$ edges will gradually isolate this community of nodes from the rest of the network nodes.

8 CONCLUSIONS

Network science continues to be a very fertile research and development area. The more our world becomes connected via 5G and Internet of Things (or Everything), the more network science evolves into a precious tool for graph-data analysis. One of the central concepts in this field, namely centralities despite counting half a century of life, is still hot giving new definitions, and new insights into the networks' organization. In this article, we introduced a new member into the family of shortest path betweenness centralities, namely sink group betweenness centrality. The purpose of this measure is to identify nodes which are in positions to monitor/control/mediate the communication towards subsets of networks nodes. We provided a simple algorithm to calculate this measure, and also extended it for node-weighted networks, and also for the case of edge betweenness. This effort is simply out first step in a long journey to develop efficient algorithms for the calculation of these measures, to analyze its distribution in real networks, to extend them to consider sink group betweenness centrality computed for a collections of network nodes, and develop techniques for approximating them.

ACKNOWLEDGMENTS

Yannis Manolopoulos would like to thank the Aristotle University of Thessaloniki, Greece, where he serves as Professor Emeritus.

REFERENCES

- [1] G. A. A. Akanmu, F. Z. Wang, and H. Chen. 2012. Introducing weighted nodes to evaluate the cloud computing topology. *Journal of Software Engineering & Applications* 5, 11 (2012), 961–969.
- [2] D. A. Bader, S. Kintali, K. Madduri, and M. Mihail. 2007. Approximating betweenness centrality. In *Proceedings of the International Workshop on Algorithms & Models for the Web-Graph (WAW)*. 124–137.
- [3] A. L. Barabasi. 2016. *Network Science*. Cambridge University Press.
- [4] A. L. Barabasi and R. Albert. 1999. Emergence of scaling in random networks. *Science* 286, 5439 (1999), 509–512.
- [5] P. Basaras, G. Iosifidis, D. Katsaros, and L. Tassioulas. 2019. Identifying influential spreaders in complex multilayer networks: A centrality perspective. *IEEE Transactions on Network Science & Engineering* 6, 1 (2019), 31–45.
- [6] P. Basaras, D. Katsaros, and L. Tassioulas. 2013. Detecting influential spreaders in complex, dynamic networks. *IEEE Computer Magazine* 46, 4 (2013), 26–31.
- [7] P. Basaras, D. Katsaros, and L. Tassioulas. 2015. Dynamically blocking contagions in complex networks by cutting vital connections. In *Proceedings of the IEEE International Conference on Communications (ICC)*. 1170–1175.
- [8] X. Bei, W. Chen, S.-H. Teng, J. Zhang, and J. Zhu. 2011. Bounded budget betweenness centrality game for strategic network formations. *Theoretical Computer Science* 412 (2011), 7147–7168.
- [9] G. Bianconi. 2018. *Multilayer Networks: Structure and Function*. Oxford University Press.
- [10] M. G. Bonchi, F. De Francisci and Riondato M. 2016. Centrality measures on big graphs: Exact, approximated, and distributed algorithms. In *Proceedings of the ACM International Conference on the World Wide Web (WWW)*. 1017–1020.
- [11] J. Borge-Holthoefer and Y. Moreno. 2012. Absence of influential spreaders in rumor dynamics. *Physical Review E* 85, 2 (2012).
- [12] U. Brandes. 2001. A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology* 25, 2 (2001), 163–177.
- [13] U. Brandes. 2008. On variants of shortest-path betweenness centrality and their generic computation. *Social Networks* 30, 2 (2008), 136–145.
- [14] M. H. Chehreghani. 2014. Effective co-betweenness centrality computation. In *Proceedings of the ACM Conference on Web Search & Data Mining (SDM)*. 423–432.
- [15] A. Cuzzocrea, A. Papadimitriou, D. Katsaros, and Y. Manolopoulos. 2012. Edge betweenness centrality: A novel algorithm for QoS-based topology control over wireless sensor networks. *Journal of Network & Computer Applications* 35, 4 (2012), 1210–1217.
- [16] S. Dolev, Y. Elovici, and R. Puzis. 2010. Routing betweenness centrality. *J. ACM* 57, 4 (2010).
- [17] S. Dolev, Y. Elovici, R. Puzis, and P. Zilberman. 2009. Incremental deployment of network monitors based on Group Betweenness Centrality. *Inform. Process. Lett.* 109 (2009), 1172–1176.
- [18] D. Easley and J. Kleinberg. 2010. *Networks, Crowds and Markets: Reasoning About a Highly Connected World*. Cambridge University Press.
- [19] M. G. Everett and S. P. Borgatti. 1999. The centrality of groups and classes. *Journal of Mathematical Sociology* 23, 3 (1999), 181–201.
- [20] M. G. Everett and S. P. Borgatti. 2009. The centrality of groups and classes. *The Journal of Mathematical Sociology* 23, 3 (2009), 181–201.
- [21] S. Fortunato. 2010. Community detection in graphs. *Physics Reports* 486, 3–5 (2010), 75–174.
- [22] S. Fortunato and D. Hric. 2016. Community detection in networks: A user guide. *Physics Reports* 659 (2016), 1–44.
- [23] L. C. Freeman. 1977. A set of measures of centrality based on betweenness. *Sociometry* 40, 1 (1977), 35–41.
- [24] R. Geisberger, P. Sanders, and D. Schultes. 2008. Better approximation of betweenness centrality. In *Proceedings of the Meeting on Algorithm Engineering & Experiments (ALENEX)*. 90–100.
- [25] M. Girvan and M. E. J. Newman. 2002. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences* 99 (2002), 7821–7826.
- [26] A. Hala, R. J. Mondragón, P. Panzarasa, and G. Bianconi. 2013. Multiplex PageRank. *PLOS One* 8, 10 (2013), e78293.
- [27] X. Huang, D. Chen, T. Ren, and D. Wang. 2021. A survey of community detection methods in multilayer networks. *Data Mining & Knowledge Discovery* 35 (2021), 1–45.
- [28] W. Hwang, T. Kim, M. Ramanathan, and A. Zhang. 2008. Bridging centrality: Graph mining from element level to group level. In *Proceedings of the ACM Conference on Knowledge Discovery & Data Mining (KDD)*. 336–344.
- [29] D. Katsaros and P. Basaras. 2015. Detecting influential nodes in complex networks with range probabilistic control centrality. In *Coordination Control of Distributed Systems*, J. H. van Schuppen and T. Villa (Eds.). Lecture Notes in Control and Information Sciences, Vol. 456. Springer-Verlag, 265–272.
- [30] D. Katsaros, N. Dimokas, and L. Tassioulas. 2010. Social network analysis concepts in the design of wireless ad hoc network. *IEEE Network Magazine* 24, 6 (2010).
- [31] M. Kitsak, L. K. Gallos, S. Havlin, F. Liljeros, L. Muchnik, H. E. Stanley, and H. A. Makse. 2010. Identification of influential spreaders in complex networks. *Nature Physics* 6 (2010), 888–893.
- [32] E. D. Kolaczyk, D. B. Chua, and M. Barthélemy. 2009. Group betweenness and co-betweenness: Inter-related notions of coalition centrality. *Social Networks* 31 (2009), 190–203.
- [33] E. D. Kolaczyk, D. B. Chua, and M. Barthélemy. 2009. Group betweenness and co-betweenness: Inter-related notions of coalition centrality. *Social Networks* 31, 3 (2009), 190–203.
- [34] A. N. Langville and C. D. Meyer. 2006. *Google's PageRank and Beyond: The Science of Search Engine Rankings*. Princeton University Press.
- [35] V. Latora, V. Nicosia, and G. Russo. 2017. *Complex Networks: Principles, Methods and Applications*. Cambridge University Press.
- [36] L. Li, G. Wang, and M. Yu. 2015. Pairwise co-betweenness for several types of network. *Journal of Networks* 10, 2 (2015), 91–98.
- [37] L. Maccari, L. Ghio, A. Guerrieri, A. Montresor, and R. Lo Cigno. 2020. Exact distributed load centrality computation: Algorithms, convergence, and applications to distance vector routing. *IEEE Transactions on Parallel & Distributed Systems* 31, 7 (2020), 1693–1706.
- [38] N. Magaia, A. P. Francisco, P. Pereira, and M. Correia. 2015. Betweenness centrality in Delay Tolerant Networks: A survey. *Ad Hoc Networks* 33 (2015), 284–305.
- [39] P. Mahendra, P. Mikhail, and H. Liaquat. 2013. Percolation centrality: Quantifying graph-theoretic impact of nodes during percolation in networks. *PLOS One* 8, 1 (2013).
- [40] N. Masuda and R. Lambiotte. 2016. *A Guide to Temporal Networks*. Series on Complexity Science, Vol. 4. World Scientific.
- [41] J. Matta, G. Ercal, and K. Sinha. 2019. Comparing the speed and accuracy of approaches to betweenness centrality approximation. *Computational Social Networks* 6 (2019).
- [42] M. E. J. Newman. 2010. *Networks: An Introduction*. Oxford University Press.
- [43] P. Pantazopoulos, M. Karaliopoulos, and I. Stavrakakis. 2014. Distributed placement of autonomic Internet services. *IEEE Transactions on Parallel & Distributed Systems* 25, 7 (2014), 1702–1712.
- [44] D. Papakostas, T. Kasidakis, E. Fragkou, and D. Katsaros. 2021. Backbones for Internet of Battlefield Things. In *Proceedings of the IEEE/IFIP Conference on Wireless On-demand Network Systems & Services (WONS)*. 116–123.
- [45] R. Puzis, Y. Elovici, and S. Dolev. 2007. Fast algorithm for successive computation of group betweenness centrality. *Physical Review E* 76 (2007), 056709.
- [46] H. S. Ramos, A. C. Frery, A. Boukerche, E. M. R. Oliveira, and A. A. F. Loureiro. 2014. Topology-related metrics and applications for the design and operation of wireless sensor networks. *ACM Trans. on Sensor Networks* 10, 3 (2014), 1–35.
- [47] G. Seyfang and N. N. Longhurst. 2013. Growing green money? Mapping community currencies for sustainable development. *Ecological Economics* 86 (2013), 65–77.

Bringing Common Subexpression Problem from the Dark to Light: Towards Large-Scale Workload Optimizations

Mohamed Kechar
LITIO, Université Oran1 Ahmed Ben
Bella, Algeria
Ecole Supérieure en Informatique,
Sidi Bel Abbès, Algeria
m.kechar@esi-sba.dz

Ladjet Bellatreche
LIAS/ISAE-ENSMA
Poitiers, France
bellatreche@ensma.fr

Safia Nait-Bahloul
LITIO, Université Oran1 Ahmed Ben
Bella
Algeria
nait-bahloul.safia@univ-oran1.dz

ABSTRACT

Nowadays large-scale data-centric systems have become an essential element for companies to store, manipulate and derive value from large volumes of data. Capturing this value depends on the ability of these systems in managing large-scale workloads including complex analytical queries. One of the main characteristics of these queries is that they share computations in terms of selections and joins. Materialized views (MV) have shown their force in speeding up queries by exploiting these redundant computations. MV selection problem (VSP) is one of the most studied problems in the database field. A large majority of the existing solutions follow workload-driven approaches since they facilitate the identification of shared computations. Interesting algorithms have been proposed and implemented in commercial DBMSs. But they fail in managing large-scale workloads. In this paper, we presented a comprehensive framework to select the most beneficial materialized views based on the detection of the common subexpressions shared between queries. This framework gives the right place of the problem of selection of common subexpressions representing the causes of the redundancy. The utility of final MV depends strongly on the selected subexpressions. Once selected, a heuristic is given to select the most beneficial materialized views by considering different query ordering. Finally, experiments have been conducted to evaluate the effectiveness and efficiency of our proposal by considering large workloads.

CCS CONCEPTS

• **Information systems** → **Data warehouses; Query optimization; Database views; Online analytical processing engines; Query optimization; Data warehouses.**

KEYWORDS

Materialized view selection, common subexpressions, multiple views processing plan, view utility

ACM Reference Format:

Mohamed Kechar, Ladjet Bellatreche, and Safia Nait-Bahloul. 2021. Bringing Common Subexpression Problem from the Dark to Light: Towards Large-Scale Workload Optimizations. In *25th International Database Engineering & Applications Symposium (IDEAS 2021)*, July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 9 pages. <https://doi.org/10.1145/3472163.3472180>

1 INTRODUCTION

Materialized views (MV) are widely used to speed up query processing thanks to their storage of the query results. A MV is associated with *redundancy* – it is redundant structure since it replicates data [1] and in the same time contributes to reducing redundant query computations [35]. MV were named for the first time in 80's in the context of OLTP databases [7]. The arrival of data warehouse technology in 90's put MV as the technique par excellence to optimize OLAP queries [12]. Recently, MV have been leveraged to optimizing large-scale workloads involving both user jobs and analytical queries [14, 35]. Two main hard problems are associated to MV: (1) MV selection problem (VSP) and (2) query rewriting in the presence of selected MV [18]. VSP consists in selecting a set of MV that optimizes a given workload. In [12], three alternatives were given to select MV: (i) materialization of whole workload (costly since it requires storage budget and maintenance overhead), (ii) materialization of nothing, and (iii) partial materialization by exploiting the *dependency* that may exist among selected views. The third scenario is considered in most of the studies related to MVP. VSP is one of the studied problems in database field.

The spectacular interest of database community to the VSP necessitates a historical analysis. The studies related to VSP passes through three main periods: (1) **Golden Age** [1990-2010], where approximately 100 papers with "materialized view selection" in the title are found in googlescholar.com published in major database conferences and journals. This golden age is characterized by the arrival of data warehouse technology and the crucial need to optimize complex OLAP queries [9]. MV have become an essential technique of the physical design phase. Due to this importance and the complexity of this selection, several research efforts from academia and industry have been conducted. Their findings have been implemented in commercial and open-source DBMSs such as Data Tuning Advisor for SQL Server, Design Advisor for DB2, SQL Access Advisor for Oracle, and Parinda for PostgreSQL DBMS [19].

In terms of capitalization on these research efforts, these findings allow the definition of a comprehensive methodology for selecting

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Association for Computing Machinery.

ACM ISBN 978-1-4503-8991-4/21/07...\$15.00

<https://doi.org/10.1145/3472163.3472180>

MVs composed of four major phases: (i) **view candidate generation**, (ii) **benefit estimation models**, (iii) **MV selection**, and (iv) **MV validation**.

- (1) The first phase consists in selecting common equivalent common subexpressions of a workload. Selecting them for materialization purposes is a hard task. This is because it depends on the join order of each query and on how individual query plans are *merged*. To perform this selection, existing studies focused on proposing graph structures such as multiple view processing plan (MVPP) [33] (and its variants multi-view materialized graph [6]) and hypergraph [8], AND-OR view graph [25], and data cube lattice [12], as support for merging individual query plans. These structures are exploited by greedy and 0-1 linear programming algorithms to select the best-unified query plan. These studies use the simplest assumptions (optimal query plan) in obtaining the individual plans and then propose algorithms to perform the merging.
- (2) To estimate the benefit of materialized a common expression in reducing the overall query processing, usually, mathematical cost models are needed to evaluate the utility of this materialization. This utility can be seen as the difference between the cost of executing workload with this expression and without it.
- (3) Since all these common expressions cannot be materialized due to the MVS constraints such as storage cost, the use of algorithms for selecting the appropriate ones is required. The survey of [20] overviews the different classes of used algorithms in selecting MVs.
- (4) Once all views are selected, the original queries are then rewritten based on these views [18].

The second period that we call the **Algorithmic Age** [2010–2018] saw the raise of the development of new algorithm classes driven by data mining [16] and game theory [3], by reproducing the initial findings related to the other three phases. The third period that we call **Big Data and Machine Learning** [2018–Now] saw the VSP getting benefit from different aspects brought from Big Data Analytics and machine and deep learning techniques that covering the availability of hardware infrastructures and new programming paradigms such as Map-Reduce and Spark. The second phase for instance has been leveraged by exploiting Map Reduce framework to select the best MVPP [5]. Recently, Machine learning techniques contribute in launching a new topic known by learned cardinality that consists in estimating intermediate results of a given query [35]. These techniques have been also used to identify the join order [34]. Recently, the requirement of optimizing large-scale workloads in the Big Data context has pushed the database community to consider the problem of subexpression selection for large-scale workloads.

By analyzing the literature, we figure out that phase 3 received much interest compared to other phases, especially in the first two periods. One of the impacts of the third period promotes all phases, with a special interest to the first one to enumerate the different view candidates for a large-scale workload, where query plans are merged in bottom-up fashion to generate the MVPP. The recent work Yuan et al. [35] is the pioneer in considering all phases

together which motivates us to consider the VSP with all its phases with a special focus on phase 3.

In this paper, we propose a global framework for VSP, by highlighting all phases in the context of Big Data Warehouses known by analytical queries involving joins, aggregations and selections on dimension tables. We claim that the first phase of VSP is a precondition for having efficient materialized views. Unfortunately, it has been hidden by phase 3 in the state of art. It should be noticed that the selection of view candidates depends on how individual query plans are generated and merged. This generation depends on the order of the joins of each query. We perform this selection, we propose two classes of algorithms: (i) *statistical-driven algorithms* and (ii) *an exhaustive algorithm*. The first class is motivated by the recent advances of period 3 in terms of using learned techniques to estimate different cardinalities. Having such estimation pushes allows us in proposing two algorithms that attempt in reducing the size of intermediate results of queries: in the first algorithm, the fact table is joined with dimension tables following their size (from small to large). *In the second algorithm, the fact table is joined with dimension tables following their FAN-OUT in ascending order.* This later represents the average number of tuples of the fact table pointed by the tuples of a dimension table satisfying the selection predicate defined on that table. An exhaustive algorithm called *ZigZag* generates all possible individual trees for each query in the workload. Once query plans in the above algorithms are obtained, a merging algorithm is then applied to select the final MVPP in an incremental way. This merging has to increase the benefit of the final selected MV. Once the final MVPP obtained an algorithm for selecting appropriate views is executed. In this study, we show the strong dependency between all phases of VSP.

The remainder of the paper is organized as follows. In Section 3, fundamental concepts and a formalisation of our studied problem are given. In Section 2, we review the most related view selection algorithms. In Section 4, we introduce our MVS Framework. Then, in Section 5, we detail the main module of our materialized view selection Framework. Section 6 details our conducted experiments considering large-scale workloads. Section 7 concludes our paper and highlights the future directions.

2 LITERATURE REVIEW

In this section, we present an overview on the major studies covering the four phases of VSP. Works developed in the Golden Age [11–13, 15, 17, 24, 28, 30, 32, 33, 36] have proposed various types of algorithms to find the best subset of materialized views from 2^m possible subsets, where m is either the number of views (inner nodes), where MVPP/And view graphs, or the number of dimensions (where lattice cubes are used). [36] introduced a hierarchical algorithm with two levels to solve the VSP. They applied evolutionary algorithm to find materialized views from the MVPP constructed using the heuristic algorithm proposed in [33]. In order to reduce the MV maintenance cost, [32] introduced a view relevance measure to determine how a candidate view fits into the already selected views and how it helps in their maintenance. [24] combined a simulated annealing algorithm and iterative improvement algorithm into a two-phase algorithm. Firstly, the two-phase algorithm improves iteratively the initial set of candidate views. The

simulated annealing algorithm tries to find the best MV starting from this initial set of candidate views.

Several studies were conducted in the Algorithmic Age period [2–4, 8, 10, 21, 27, 31]. Based on game theory concepts, [3] developed a greedy based algorithm. They modeled the VSP as a game with two opposite players: the query processing cost player and the view maintenance cost player. One player selects views with a high processing cost and the second one selects views with a high maintenance cost. Thus, at game-ending, the remaining views having a lower query processing cost and a lower view maintenance cost are appropriate for materialization. [27] extended the algorithm proposed in [3] to an evolutionary version with populations of players. After applying a succession of genetic operations (selection, mutation, and crossover) on populations of players, the extended algorithm finds a better solution than in the case of two players. Inspired by techniques used in the electronic design automation domain, [8] modelled the global processing plan of queries by hypergraph, in which the joins shared between a set of queries are edges. Then, they proposed a deterministic algorithm to select the best set of views to materialize from the MVPP generated from the hypergraph.

Regarding the Big Data and Machine Learning period, [5, 14, 22, 35] focused mainly on the second phase and proposed cost estimation models using deep learning models to estimate more accurately the benefit of the materialized views. These cost models help in selecting the best materialized views that contribute in reducing the overall query processing.

The aforementioned MV algorithms proposed in different periods select the best MV from one MVPP constructed by merging only the optimal query trees for queries. However, the construction of one MVPP from optimal query trees does not guarantee its optimality [26, 36]. Consequently, other MVPP constructed from merging other query trees can provide more beneficial MV. Moreover, almost of the reviewed algorithms, materialize additional views to decrease the maintenance cost of other materialized views. Although the decrease of the view maintenance cost, those additional views increase the required storage space.

Motivated by this fact, in this paper, we propose a MV framework to select the most beneficial materialized views based on the detection of the common subexpressions shared among queries. Unlike existing algorithms, our framework: (1) manages large-scale workloads, and (2) uses several query trees to generate several MVPP looking for the best MV composed of a less number of views that optimize several queries.

3 PROBLEM FORMULATION

3.1 Definitions

- **A Query tree** can be represented by a labeled directed acyclic graph (DAG) that describes the processing sequence of relational operations (e.g. selection, projection, join) that produces the result of a query (see Figure 1(a)). Leaves nodes are labeled by the base relations of the database schema required to process the query and the root node is the final result of the query labeled by Q , while the inner nodes are relational operations.

- **A multiple views processing plan MVPP** is a labeled DAG, which provides a manner to answers the N queries in the workload W against the data warehouse. Given N queries, an MVPP is constructed by merging incrementally their respective N query trees; one for each query, by sharing nodes defining common subexpressions among queries (see Figure 1(b)). Leaf nodes in the MVPP are the base relations and the roots nodes are queries, while inner nodes are relational operations.

3.2 Problem statement

Given a Data Warehouse schema and a workload composed of star queries $W = \{Q_1, Q_2, \dots, Q_n\}$, our aim is to generate a set of materialized views $MV = \{v_1, v_2, \dots, v_z\}$ based on common subexpressions, such that the processing cost of W is optimized when using views in MV to answer queries in W .

The problem of common subexpressions detection is generally based on the query tree merging mechanism. It is more challenging for a large-scale workload having: (1) a large number of queries, and (2) a high number of query trees per query. To illustrate the complexity of this problem, we force our-self to follow an *incremental presentation* of its complexity.

Let Q_i be a star join query from W involving a fact table and x_i dimension tables. The number of all possible individual trees corresponding to Q_i , denoted by IDQ_i is given by the following equation:

$$IDQ_i = 2 \times (x_i - 1)! \quad (1)$$

This formula supposes that left-deep join strategy is used [29] to perform this query.

Let us assume that the queries of the workload are *ordered*. To generate the optimal MVPP for view materialization obtained from merging individual plans is needed. This generation requires the following complexity:

$$\text{ordered_nb_merging} = \prod_{i=1}^N IDQ_i \quad (2)$$

In the case that queries of the workload are *not a priori ordered*, this increases dramatically the complexity as shown in the following equation:

$$\text{no_ordered_all_merging} = N! \times \prod_{i=1}^N IDQ_i \quad (3)$$

This high complexity requires intelligent algorithms to identify the most beneficial individual plan merging for view materialization.

4 FRAMEWORK OVERVIEW

In this section, we propose a comprehensive framework that leverages the findings of the two VSP periods (Golden and Algorithmic Ages). Our motivation is to bring the problem of common subexpressions from the dark to the light. Figure 2 illustrates our framework. It addresses the common subexpressions detection and views materialization by considering three important correlated aspects: (i) the definition of individual query plans, (ii) the establishment of query ordering, and (iii) the identification of the most beneficial common subexpressions for materialization purpose.

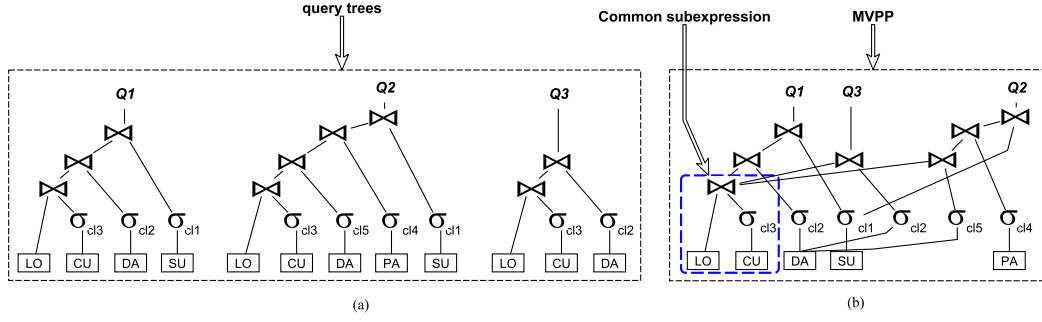


Figure 1: (a) Query trees, (b) Merging query trees in MVPP

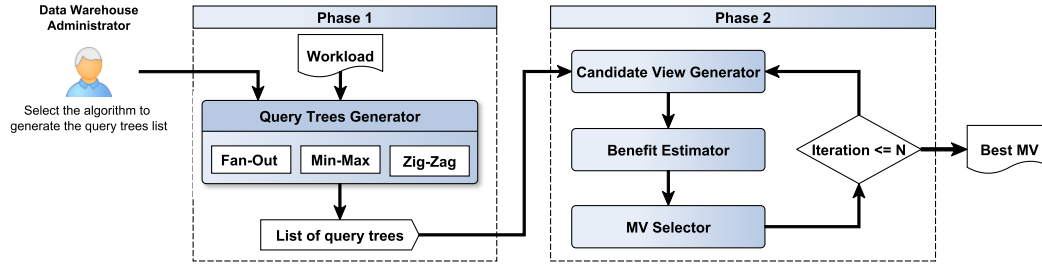


Figure 2: MVS Framework overview

- (1) To address the first aspect (**Query Trees Generator**), we propose three algorithms to generate the best individual tree for each query. The two first algorithms are statistic-based since they use the Fan-Out and Min-Max criteria to generate only one individual tree for each query. Since statistical information is critical and sometimes difficult to have before deploying the target data warehouse, exploratory algorithms are required. From this perspective, we propose a Zig-Zag algorithm that generates all possible query trees for each query in W .
- (2) To address the second aspect, we adopt a heuristic technique to investigate a reduced number of query ordering. The heuristic examines N query ordering. Starting from an initial query order, (a) it the corresponding query trees merging, (b) detects the common subexpressions, (c) generates their corresponding MV, and (d) estimates their benefit. This process is repeated N iterations by moving the query at the first position in the current query order to the last position. **At each iteration, the MV Selector in charge of selecting views, identifies the best MV that maximizes the benefit.**
- (3) Regarding the third aspect, given a query order with the query trees for each query, the challenge is to detect the common subexpressions shared among queries. To address this challenge, we propose an incremental query tree merging algorithm that groups queries using the same subexpression

into a partition. Each detected subexpression is a view candidate (that will be managed by the **Candidate View Generator**). Since it is more practical to generate less number of materialized views leading to optimizing the performance of numerous queries in the workload, our incremental query trees merging aims at finding less number of common subexpressions (managed by the **Candidate MV Estimator**).

5 SUBEXPRESSION IDENTIFICATION FOR MATERIALIZATION

In this section, we show the strong connection between common subexpression selection and view materialization. We then outline the main generic components of our framework which are: query tree generator, candidate views generator, candidate MV estimator, and MV selector. Each component is associated to one or several algorithms (Figure 2).

5.1 Query trees generator

For each query $Q_i \in W$, we extract respectively its base tables and its selection predicates. Then, we generate for each query a list of query trees according to the applied algorithm, namely, Fan-Out, Min-Max, or ZigZag. The output of the query generator module is a list of queries with their respective lists of query trees.

- (1) **Fan-Out algorithm.** For each query $Q_i \in W$, and for each clause cl_i of selection predicates used to filter a dimension

table T_i in Q_i , we compute the Fan-Out value of cl_i according to the Formula 4.

$$\text{Fan-Out}(cl_i) = \frac{\| \text{FactTable} \bowtie \sigma_{cl_i}(T_i) \|}{\| \text{FactTable} \|} \quad (4)$$

where $\| \cdot \|$ and \bowtie represent respectively the number of instances, and semi-join operation.

Then, we generate one query tree for Q_i by ordering the join of the base tables according to the ascending order of the Fan – Out values.

Example 5.1. Let us consider the query Q shown in Figure 3(a). This query contains three tables: *Lineorder* that plays the role of a fact table (having 1000 tuples), and *DWdate*, and *Customer* playing the role of dimension tables. Two selection predicates are defined on these dimension tables: $\text{dwdate.d_year} = '1995'$ (cl_1) and $\text{customer.c_region} = 'ASIA'$ (cl_2). In this case, we have to compute two FAN-OUT corresponding to these predicates (cl_1 and cl_2). This computation can be done using SQL statements illustrated in Figures 3(b) and 3(c). Of course, this calculation requires access to the database. Otherwise, the use of statistics is desired as for other parameters such as selectivity factors, the size of intermediate results.

If $\text{Fan-Out}(cl_1) < \text{Fan-Out}(cl_2)$, the join order of the query Q is $LO \bowtie DA \bowtie CU$, otherwise $LO \bowtie CU \bowtie DA$. Figures d and e show their individual trees.

- (2) **Min-Max algorithm.** For each query $Q_i \in W$, we generate one query tree by ordering the join of the base table according to their size from minimum size to maximum size.
- (3) **ZigZag algorithm.** Contrary to the first two algorithms, it does not assume any a priori join order. So for a given query Q_i accessing x_i tables, the ZigZag generates all $2 \times (x_i - 1)!$ query trees.

As we can see from Figure 4, the query generator enumerates three query trees: one for each query using Fan-Out/Min-Max algorithms. When the Zig-Zag algorithm is used, the query generator has to consider respectively for Q_1 , Q_2 , and Q_3 : $2 \times (4-1)! = 12$, $2 \times (5-1)! = 48$, and $2 \times (3-1)! = 4$ query trees (cf. Figure 4).

5.2 Candidate views generator

This module considers the different query trees generated by our above algorithms and aims at merging them in to increase the sharing degree among queries. This implies query ordering. At the end, queries with common subexpressions are grouped in a partition. More concretely, this generator consists in dividing the initial queries into K partitions, where each one contains either shared queries or isolated ones.

Let L be a list containing trees of all queries of the workload and indexed by queries. This generator has two main steps:

- (1) **Initialization of Partitions:** To generate our partition, we start with the first two queries of our list and we try to merge their query trees. Note that the merging is possible if a common node exists (selection or join). If a common node exists, then a partition containing these two queries is created. Otherwise, a partition is created for each query.

- (2) **Updating Partitions:** The above steps will generate either one or two partitions. This step has to explore the rest of the queries. In the case where the third query shares common node queries in an initial partition, then it will be included in that partition. Otherwise, a new partition is created. This process is then repeated till the end of the list.

Example 5.2. To illustrate this step, let us consider an example of three queries (Q_1 , Q_2 , Q_3), where the FAN-OUT algorithm is used. Recall that our list will contain three query trees as shown in Figure 4. We have to check whether Q_1 and Q_2 are *mergeable*. Since these two queries share a common node ($LO \bowtie \sigma_{cl_1}(SU)$), a partition P_0 is then created. This partition corresponds to the first MVPP (MVPP₀). After that, we have to examine the third query that does not share any node of the MVPP₀. As a consequence a new partition P_1 is created containing Q_3 .

- (3) **Candidate views generation:** For each partition P_i (of the above steps) with more than two queries, we generate a candidate view v_i corresponding to the node used to merge these queries.

Example 5.3. The view v_1 shown in Figure 5(a) is a candidate view generated from the merging of the query trees given by the Query Trees Generator using the ZigZag algorithm (see Figure 4). The view v_1 is defined by the subexpression $LO \bowtie \sigma_{cl_13}(CU)$.

The three steps are re-executed by changing the initial order of the queries. The query order impacts the common subexpression detection. Since, the consideration of all query order requires $N!$, we propose a query ordering in a round robin fashion (which requires N orderings).

5.3 Candidate materialized views estimator

In the above steps, a list of view candidates is obtained. This module has to quantify the benefit/utility of each candidate in reducing the overall query cost. To do so, the initial queries of our workload have to be rewritten by considering these candidates.

An example of query rewriting is shown in Figure 5(b), where queries Q_1 , Q_2 , Q_3 of the partition P_0 (see Figure 4 which is the case of the ZigZag algorithm) are rewritten using v_1 .

More formally, using each view candidate v_i , we estimate the processing cost $QPC(Q_j, v_i)$ for each query $Q_j \in P_i$. The $QPC(Q_j, v_i)$ is equal to the query processing of the equivalent rewritten query Q'_j on v_i . Finally, we compute the overall query processing cost of W in the presence of MV by the Formula 5.

$$QPC(W, MV) = \underbrace{\sum_{i=1; Q_j \in P_i, v_i \in MV}^{K'} QPC(Q_j, v_i)}_{\text{optimized queries}} + \underbrace{\sum_{i=1; Q_j \in P_i}^{K''} QPC(Q_j)}_{\text{not optimized queries}} \quad (5)$$

where K' , K'' are respectively the number of partitions with more than two queries sharing common-subexpressions (i.e. partitions that group optimized queries), and the number of partitions of only one query (i.e. not optimized queries). Using the $QPC(W, MV)$, we define the Formula 6 to compute the first metric to measure the

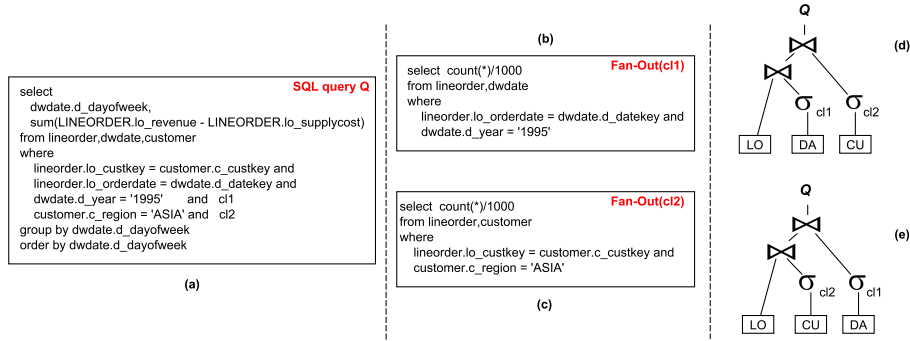


Figure 3: Query trees generator: case of Fan-Out

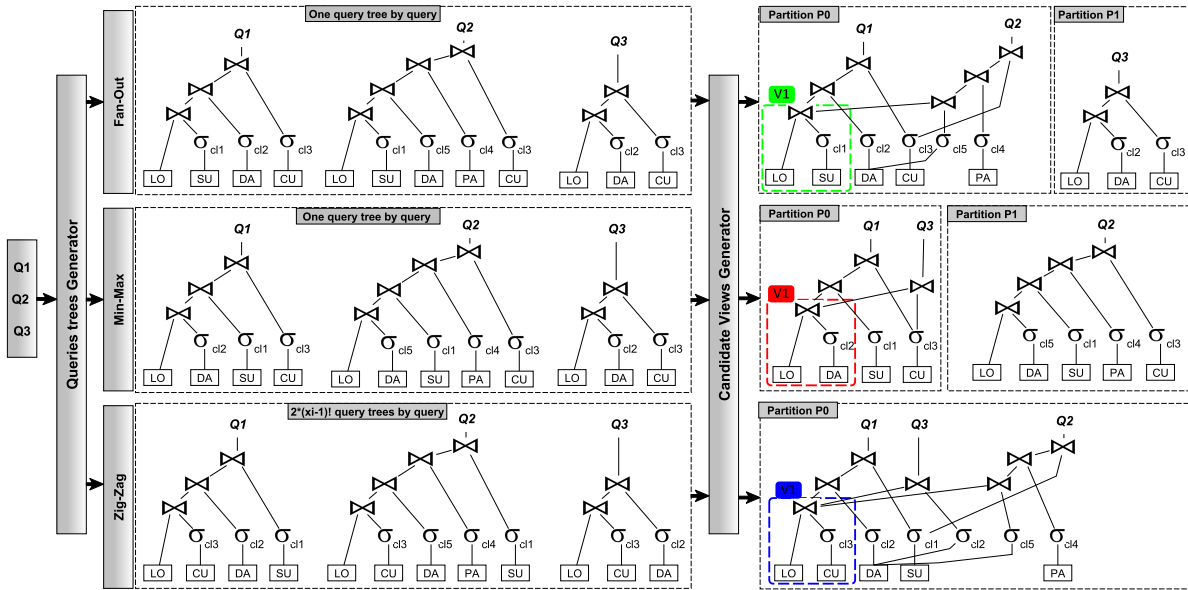


Figure 4: Illustration of different components of our framework

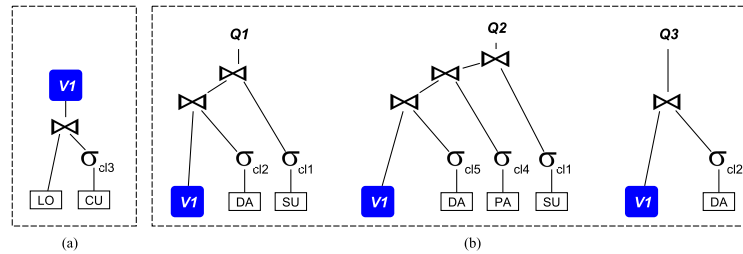


Figure 5: Candidate view generation and queries rewriting

benefit of a set MV of candidate materialized view.

$$\text{Benefit}(MV) = \text{QPC}(W) - \text{QPC}(W, MV) \quad (6)$$

where $\text{QPC}(W)$ represents the query processing of the workload W without any materialized views. To select wisely the best MV , we define a second metric that measures the quality of the set MV

defined by equation 7.

$$\text{Quality(MV)} = \frac{\text{Number_Optimized_Queries}}{\text{Number_Views}} \quad (7)$$

where `Number_Optimized_Queries` is the number of optimized queries resulting from the query grouping, and the `Number_Views` is the number of candidate views, which is equal to K' . We recall here that in the computation of the QPC, we used a cost model based on the selection cardinality estimation, and join cardinality estimation formulas. Table 1 resumes the values of $\text{QPC}(W)$, $\text{QPC}(W, MV)$, and the $\text{Benefit}(W, MV)$ expressed in terms of Inputs/Outputs (I/O) unit. For lack of space, other values as the cost of creation of views and their size are not shown in the Table 1. As we can see, for 3 queries, the candidate view v_1 generated from the merging of the Zig-Zag query trees (see Figure 4, the view with blue color) provides better benefit (value with the blue color in Table 1) than other views. This is mainly due to the fact that the detected subexpression, if materialized, will have utility for the three queries. But, as we can see, whatever the algorithm used to generate the query trees, the candidate views generator module is able to detect common sub-expression between the three queries, such that their materialization optimizes significantly the workload.

5.4 Materialized views selector

Since we have view candidates for each query order, therefore, we need a mechanism selecting the best views. This selection will be performed using our cost models.

In the next section, we outline the extensive experiments conducted in order to show the effectiveness of the proposed framework.

6 EXPERIMENTAL STUDY

To study the effectiveness of the proposed algorithms, we conducted intensive experiments using workloads of different sizes. All experiments are conducted under a machine with Intel processor Core i5 2.9GHz, 8GB of RAM, and 500 GB hard disk. In all experiments, we used the well-known Star Schema Benchmark (SSB) [23] as a baseline for the data warehouse schema and for the datasets. Under Oracle12c¹ database management system, we created physically a data warehouse DW having the schema tables: the fact table *LINEORDER* of size 102M rows (10 GB), and the dimension tables: *CUSTOMER*, *SUPPLIER*, *PART*, and *DATE*, with the respective sizes in rows, 3M, 200K, 1.4M, and 2566.

To evaluate the performance and the scalability of our different algorithms associated to the components of our framework, we consider 10 workloads with the following queries: 30, 60, 90, 130, 160, 200, 250, 300, 700, and 1000 queries. The results of the experiments are discussed hereafter. Figure 6a shows both the query processing costs without and with MV selected by Fan-Out/HA, Min-Max/HA, and Zig-Zag/HA. Figure 6b shows the improvement rates in query processing cost provided by each algorithm for each workload. The first lesson from these experiments is that the selected MV by our algorithms significantly improve the overall query processing. This confirms the main role of MV for speeding up OLAP queries

The second interesting lesson is related to the correlation between the size of the workload and query performance. As the size increases, the improvement rates of the query processing also increase (Figure 6b). This shows the strong connection between common subexpression and view selection problems. When the number of queries increases the probability of sharing is high which increases the number of view candidates.

For instance, for the workload with 1000 queries, the Fan-out/HA, Min-Max/HA, and ZigZag/HA provide respectively improvement rates of 88,97%, 78,39%, and 83,15% in the query processing cost. These algorithms detect respectively 216, 161, 143 common subexpressions. The materialization of these views optimizes respectively 901, 813, 856 queries among 1000 queries, which represent a high performance.

Based on this above result, we remark that the ZigZag/HA can detect the most beneficial subexpressions for materialization. This is due to its exploratory search. This is confirmed by the improvement rates result shown in Figure 6b. From 30 queries up to 300 queries, the ZigZag/HA outperforms Fan-Out/HA and Min-Max/HA. However, when the workload increases up to 1000 queries the Fan-Out/HA algorithm performs better than Min-Max/HA and Zig-Zag/HA.

These results were predictive when we presented the complexity of merging process in Section 3.2. The ZigZag/HA needs large runtime to explore large number of query trees merging looking for the most beneficial subexpressions.

If we examine the three algorithms in terms of the quality metric computed by the Formula 7, and whose values are plot in Figure 7a, we can easily conclude that Min-Max/HA and Zig-Zag/HA are more effective than Fan-Out/HA.

Also, we can conclude that for moderate workload Min-Max/HA is better than Zig-Zag/HA in terms of effectiveness. But, for a large workload, Zig-Zag/HA provides a good compromise between the number of optimized queries and the number of selected materialized views. Finally, if the materialized views selection is constrained by the space storage limit, in this case, the Fan-Out/HA algorithm is the best one. As shown in Figure 7b, whatever the size of the workload, Fan-Out/HA is able to select a set of materialized views that improves the query processing cost and requires less storage space.

To summarize our result, we can conclude that all three algorithms are interesting alternatives for VSP. They can be applied according to the decision-support application requirements and to the structure of the workload. They can be merged by dividing workload queries based on their importance. For each class, a specific algorithm can be executed. Min-Max/HA and FAN-OUT/HA certainly offer interesting performances, but it must be said that they are highly dependent on statistics. Whereas, the Zig-Zag/HA for moderate workloads represents an interesting alternative.

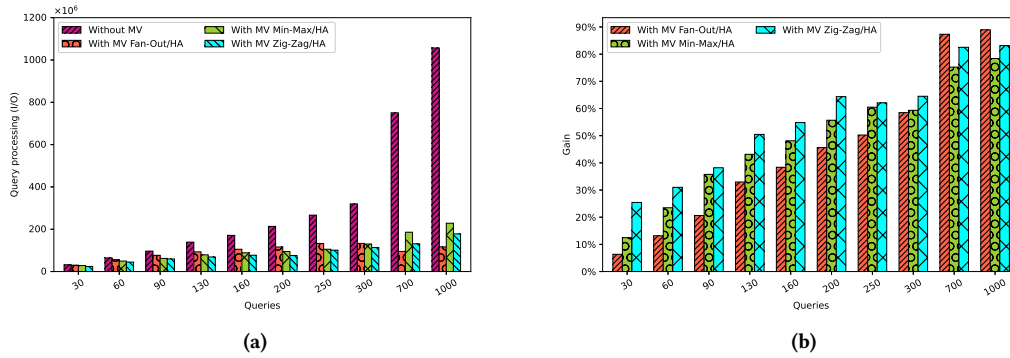
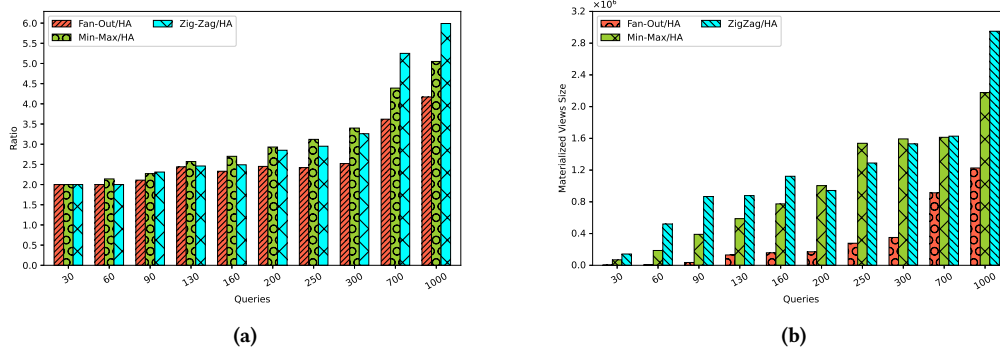
7 CONCLUSION

Nowadays decision-support applications tend to use large-scale workloads executing on data warehouses with various tables. This situation pushes researchers to revisit the existing findings to optimize such workloads. Among these findings, we can cite the case of materialized views. These techniques if well used can contribute

¹<https://www.oracle.com/database/12c-database/>

Table 1: Costs estimation in number of I/O

	QPC	QPC with MV ^{Fan-Out}	QPC with MV ^{Min-Max}	QPC with MV ^{Zig-Zag}
Q1	1087702	56175	229808	363608
Q2	1255439	1255439	226496	308489
Q3	1104043	72516	1104043	380807
W	3447184	1384130	1560347	1052904
Benefit		2063054	1886837	2394280
Improvement rate %		59,85%	54,74%	69,46%
Optimized queries		Q1,Q3	Q1, Q2	Q1,Q2,Q3

**Figure 6: (a) Query processing cost, (b) Query processing improvement rate****Figure 7: (a) optimized queries on Materialized views, (b) Materialized views size**

in reducing redundant computations detected in cloud queries user jobs. The problem of selecting these views is a hard problem. It has been widely studied in the traditional data warehouses, and existing studies consider workloads with few ten queries. Recently it resurfaces. This situation obliges us to elaborate the historical overview of materialized view selection problem, where three main periods have been identified: *Golden Age* [1990-2010], *Algorithmic Age* [2010-2017] and *Big Data and Machine Learning* [2018-Now]. We believe that this historical presentation will help young researchers in understanding this crucial problem. In addition to this historical review, we succeed in showing the strong connection between view materialization and common query subexpressions. In the literature,

the problem of common subexpression selection got less attention compared to other surrounding problems of VSP. Therefore, we attempt to bring it from the dark to the light. This passed by an incremental analysis of its complexity and the presentation of a framework that combines these two problems. This framework contains four components: (1) query tree generator, (2) candidate views generator, (3) candidate MV estimator, and (4) MV selector. Two classes of algorithms are given for generating individual query plans: two called Min-Max and FAN-OUT based on statistics and an exploratory algorithm called Zig-Zag. An algorithm for merging these plans is also given accompanied by a materialized view selection algorithm. These algorithms aim at increasing the utility

of the final views in terms of the reduction of the overall query performance. Several experiments were conducted to evaluate our proposal. The obtained results show the complementarity of our algorithms.

Currently, we are developing a wizard supporting our framework and studying the possibility to combine these algorithms for large-scale workload by partitioning it. Another issue concerns the consideration of new alternatives of query ordering.

REFERENCES

- [1] S. Agrawal, S. Chaudhuri, and V. R. Narasayya. Automated selection of materialized views and indexes in SQL databases. In *Vldb*, pages 496–505, 2000.
- [2] K. Aouiche and J. Darmont. Data mining-based materialized view and index selection in data warehouses. *J. Intell. Inf. Syst.*, 33(1):65–93, 2009.
- [3] H. Azgomi and M. K. Sohrabi. A game theory based framework for materialized view selection in data warehouses. *Eng. Appl. Artif. Intell.*, 71:125–137, 2018.
- [4] H. Azgomi and M. K. Sohrabi. A novel coral reefs optimization algorithm for materialized view selection in data warehouse environments. *Appl. Intell.*, 49(11):3965–3989, 2019.
- [5] H. Azgomi and M. K. Sohrabi. Mr-mvpp: A map-reduce-based approach for creating mvpp in data warehouses for big data applications. *Information Sciences*, 570:200–224, 2021.
- [6] X. Baril and Z. Bellahsene. Selection of materialized views: A cost-based approach. In *CAiSE*, pages 665–680, 2003.
- [7] J. A. Blakeley, P. Larson, and F. W. Tompa. Efficiently updating materialized views. In *ACM SIGMOD*, pages 61–71, 1986.
- [8] A. Boukorca, L. Bellatreche, S. A. B. Senouci, and Z. Faget. Coupling materialized view selection to multi query optimization: Hyper graph approach. *Int. J. Data. Warehous. Min.*, 11(2):62–84, 2015.
- [9] S. Chaudhuri and V. R. Narasayya. Self-tuning database systems: A decade of progress. In *Vldb*, pages 3–14, 2007.
- [10] A. Gosain and K. Sachdeva. Selection of materialized views using stochastic ranking based backtracking search optimization algorithm. *Int. J. Syst. Assur. Eng. Manag.*, 10(4):801–810, 2019.
- [11] H. Gupta. Selection of views to materialize in a data warehouse. In F. Afrati and P. Kolaitis, editors, *ICDT*, pages 98–112, 1997.
- [12] V. Harinarayan, A. Rajaraman, and J. D. Ullman. Implementing data cubes efficiently. In *ACM SIGMOD*, pages 205–216, 1996.
- [13] Jeffrey Xu Yu, Xin Yao, Chi-Hon Choi, and Gang Gou. Materialized view selection as constrained evolutionary optimization. *IEEE Trans. Syst. Man Cybern. Part C: Appl. Rev.*, 33(4):458–467, 2003.
- [14] A. Jindal, K. Karanasos, S. Rao, and H. Patel. Selecting subexpressions to materialize at datacenter scale. *Proc. VLDB Endow.*, 11(7):800–812, Mar. 2018.
- [15] Jorng-Tzong Horng, Yu-Jan Chang, Baw-Jhiune Liu, and Cheng-Yan Kao. Materialized view selection using genetic algorithms in a data warehouse system. In *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, volume 3, pages 2221–2227 Vol. 3. IEEE, 1999.
- [16] T. V. Kumar, A. Singh, and G. Dubey. Mining queries for constructing materialized views in a data warehouse. In *Advances in Computer Science, Engineering & Applications*, pages 149–159. Springer, Berlin, Heidelberg, 2012.
- [17] M. Lee and J. Hammer. Speeding up materialized view selection in data warehouses using a randomized algorithm. *Int. J. Coop. Inf. Syst.*, 10(3):327–353, 2001.
- [18] C. Li. Rewriting queries using views. In L. LIU and M. T. ÖZSU, editors, *Encyclopedia of Database Systems*, pages 2438–2441. Springer US, Boston, MA, 2009.
- [19] C. Maier, D. Dash, I. Alagiannis, A. Ailamaki, and T. Heinis. PARINDA: an interactive physical designer for postgresql. In *EDBT*, pages 701–704, 2010.
- [20] I. Mami and Z. Bellahsene. A survey of view selection methods. *ACM Sigmod Record*, 41(1):20–29, 2012.
- [21] I. Mami, R. Coletta, and Z. Bellahsene. Modeling view selection as a constraint satisfaction problem. In *DEXA*, page 396–410, 2011.
- [22] L. Ordonez-Ante, G. Van Seghbroeck, T. Wauters, B. Volckaert, and F. De Turck. A workload-driven approach for view selection in large dimensional datasets. *J. Netw. Syst. Manag.*, pages 1–26, 2020.
- [23] P. E. O’Neil, E. J. O’Neil, and X. Chen. The star schema benchmark (SSB), 2009.
- [24] J. Phuboon-ob and R. Auepanwiriyaikul. Selecting materialized views using two-phase optimization with multiple view processing plan. *Int. J. Econ. Manag. Eng.*, 1(3):53–58, 2007.
- [25] N. Roussopoulos. The logical access path schema of a database. *IEEE Trans. Softw. Eng.*, 8(6):563–573, Nov. 1982.
- [26] T. K. Sellis. Multiple-query optimization. *ACM Trans. Database Syst.*, 13(1):23–52, Mar. 1988.
- [27] M. K. Sohrabi and H. Azgomi. Evolutionary game theory approach to materialized view selection in data warehouses. *Knowl.-Based Syst.*, 163:558 – 571, 2019.
- [28] X. Song and L. Gao. An ant colony based algorithm for optimal selection of materialized view. In *International Conference on Intelligent Computing and Integrated Systems*, pages 534–536. IEEE, Oct 2010.
- [29] M. Steinbrunn, G. Moerkotte, and A. Kemper. Heuristic and randomized optimization for the join ordering problem. *VLDB J.*, 6(3):191–208, Aug. 1997.
- [30] X. Sun and Z. Wang. An efficient materialized views selection algorithm based on pso. In *2009 International Workshop on Intelligent Systems and Applications*, pages 1–4. IEEE, May 2009.
- [31] Z. A. Talebi, R. Chirkova, and Y. Fathi. An integer programming approach for the view and index selection problem. *Data Knowl. Eng.*, 83:111–125, 2013.
- [32] S. R. Valluri, S. Vadapalli, and K. Karlapalem. View relevance driven materialized view selection in data warehousing environment. *Aust. Comput. Sci. Commun.*, 24(2):187–196, Jan. 2002.
- [33] J. Yang, K. Karlapalem, and Q. Li. Algorithms for materialized view design in data warehousing environment. In *Vldb*, pages 136–145, 1997.
- [34] X. Yu, G. Li, C. Chai, and N. Tang. Reinforcement learning with tree-lstm for join order selection. In *ICDE*, pages 1297–1308, 2020.
- [35] H. Yuan, G. Li, L. Feng, J. Sun, and Y. Han. Automatic view generation with deep learning and reinforcement learning. In *ICDE*, pages 1501–1512, 2020.
- [36] C. Zhang, X. Yao, and J. Yang. An evolutionary approach to materialized views selection in a data warehouse environment. *IEEE Trans. Syst. Man Cybern. Part C: Appl. Rev.*, 31(3):282–294, Aug 2001.

Colonization of the Internet

Bipin C. Desai*
BipinC.Desai@concordia.ca
Concordia University
Montreal, Canada



The Salt March

ABSTRACT

The internet was introduced to connect computers and allow communication between these computers. It evolved to provide applications such as email, talk and file sharing with the associated system to search. The files were made available, freely, by users. However, the internet was out of the reach of most people since it required equipment and know-how as well as connection to a computer on the internet. One method of connection used an acoustic coupler and an analog phone. With the introduction of the personal computer and higher speed modems, accessing the internet became easier. The introduction of user-friendly graphical interfaces, as well as the convenience and portability of laptops and smartphones made the internet much more widely accessible for a broad swath of users. A small number of newly established companies, supported by a large amount of venture capital and a lack of regulation have since established a stranglehold on the internet with billions of people using these applications. Their monopolistic practices and exploitation of the open

nature of the internet has created a need in the ordinary person to replace the traditional way of communication with what they provide: in exchange for giving up personal information these persons have become dependent on the service provided. Due to the regulatory desert around privacy and ownership of personal electronic data, a handful of massive corporations have expropriated and exploited aggregated and disaggregated personal information. This amounts, we argue, to the colonization of the internet.

CCS CONCEPTS

• General and reference; • Software and its engineering; • Social and professional topics; • Applied computing; • Security and privacy;

KEYWORDS

Privacy, security, smartphone, contact tracking, big tech, surveillance

ACM Reference Format:

Bipin C. Desai. 2021. Colonization of the Internet. In *25th International Database Engineering & Applications Symposium (IDEAS 2021)*, July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 11 pages. <https://doi.org/10.1145/3472163.3472179>

1 INTRODUCTION

The internet was introduced to connect computers and allow communication between these computers. With the introduction of portable personal computers and higher speed modems, access to the internet became easier. The introduction of X-windows, a graphical interface[107] and hardware incorporating such graphical interfaces in closed systems

*Corresponding Author

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Association for Computing Machinery.

ACM ISBN 978-1-4503-8991-4/21/07...\$15.00

<https://doi.org/10.1145/3472163.3472179>

brought in more users, including a cult of users of a brand and who have remained attached to this closed system and its new models! The world wide web, developed on a machine incorporating a version of this graphical interface in the mid 1990s, was followed by a graphical browser along with laptops and smartphones, making it possible for a large number of users to connect to the internet. Private capital, with tacit support of the USAian¹ government, was able to nourish the emergence of big tech: private capital was on-side since there were no regulations and no application of the existing regulations and laws to the internet. This free-for-all meant that great fortunes could be reaped and existing boundaries of acceptable practice ignored by the 'platform' designation of big-tech. The other problem was of course the lack of imagination of complacent management of existing corporations to provide the additional services. Instead of preventing monopolies in the new digital world, legislators promoted them to foster innovation at the expense of privacy and security. This was what prompted capital to be made available to the emerging robber barons of the late 20th century. These corporations headed by buccaneers started putting down their own rules and bought politicians. Their big purses allowed them to bend most politicians and anyone with independent thought and ideas was put down by the anti-populist forces[43].

These newly established companies, supported by a large amount of venture capital and lack of regulations and/or not knowing how to apply the regulations in the new digital world when services were 'free'. These new companies in the tech field have since established a stranglehold on the internet and essentially colonized it by expropriating the personal data of the users of the free service users and mining this data to exploit and influence them in subtle ways. They have exploited the opportunity and created a need in the ordinary person to replace the traditional way of communication with what they provide: these persons have become dependent on the service provided in exchange for giving up personal information. The internet, mobile phone technology, and the web, have been exploited by new companies since the original existing players in place were restricted by legislation or mostly inertia and lack of foresight. For example, the national postal services should have been called in to provide email service to supplement traditional mail service. The lack of politicians with any foresight, savvy and/or political will and the resistance to providing funds to the existing systems such as the postal service to build up expertise and infrastructure in this new domain meant that this did not occur anywhere. Some of these new tech companies, extending and scaling their infrastructure have set up cloud services(time sharing with less control). Such cloud services are tempting businesses to move their computing to such clouds and abandoning their existing infrastructure. Examples are the migration of operations such as organizational email systems, administrative services and so on to the cloud. The result is not necessarily

an improvement nor economical. One example of a colossal fiasco was the migration of a payroll software system by the government of Canada to a untested system called Phoenix: 'As a result of world-class project mismanagement on the Phoenix project, the Canadian government now owns and operates a payroll system "that so far has been less efficient and more costly than the 40-year-old system it replaced."' [82].

As in the past, the introduction of new technology has upset the status quo. Opportunities have been missed by established players. Industries such as the taxi industry have also suffered with the introduction of instant communication and location broadcasting mobile phones. New players, falsely claiming to be ride-sharing, have carved out a large chunk of the taxi business. Companies and individual operators, who paid a high price for a taxi permit, were left holding the bag. New players, breaking and challenging established regulations and using communication technology along with willing drivers with automobiles, were able to offer an alternate system and take-over a sizable chunk of the taxi business everywhere. Again the existing taxi system, with its regulation, did not see an opportunity to use the new technology. Also, they were restricted from charging different amount for the same ride as is the case with the so-called ride sharing system which could charge a rate depending on the demand, time of the day etc. since there was no regulation for the ride sharing system.

2 COLONIZATION

Throughout human existence, tribes have moved from one pasture to another. In pre-historic days, it is likely that there were no other humans in the new pasture and if there were any, the existing population would either be annihilated, absorbed by the new herd or the new herd be assimilated into or driven out. The new worlds were invaded by hordes from the European countries. Having better weapons and using the divide and conquer strategy time and again, these invaders (settlers) to the new world were able to overpower the existing population. Unfortunately, the pre-historic techniques are still being used to-date in some parts of the world[42, 109]. The practice of annihilation, dispossession and driving out was gradually replaced by the strategy of forceful conversion[71–73]:

"For over a century, the central goals of Canada's Aboriginal policy were to eliminate Aboriginal governments; ignore Aboriginal rights; terminate the Treaties; and, through a process of assimilation, cause Aboriginal peoples to cease to exist as distinct legal, social, cultural, religious, and racial entities in Canada. The establishment and operation of residential schools were a central element of this policy, which can best be described as "cultural genocide." ... Cultural genocide is the destruction of those structures and practices that allow the group to continue as a group.

States that engage in cultural genocide set out to destroy the political and social institutions of the targeted group. Land is seized, and populations are forcibly transferred and their movement is restricted. Languages are banned. Spiritual

¹USAian is pronounced U-asian; it is a more appropriate term than American since USA is but one country in N & S Americas!

leaders are persecuted, spiritual practices are forbidden, and objects of spiritual value are confiscated and destroyed. ...

In its dealing with Aboriginal people, Canada did all these things.” [39, 71]

The practice of assimilation was carried out by successive governments[74] including those under a prime minister who was awarded the peace prize[62] but did nothing to solve the problems at home! The shameful practice of abducting children continued well to the end of the 20th century[54, 72]. The same strategy was used in the USA as reported in a recent article[37].

Similar practices were present in all the, so called, new world which included N & S America and Australia and to some extent Africa. However, the topic of this paper being the colonization of the Internet, we will not dwell on this any further.

2.1 Trading Colonization

The invasion of the new world and its colonization was substituted by another type of colonization that started with trading by a number of East India Companies with various European bases[96–102]. They were established to trade in spices and other resources from the Indian subcontinent and the orient. These trading companies requiring the protection of their territory initially used the company’s hired armed man[103] which was followed by the armed forces of the company’s home country[104]. Since the ‘invaded’ country had a civilization older than the colonizing one, and had a large population these trading nations were not able to annihilate the existing population. However, just as in the new worlds, using a divide and conquer strategy, the existing system of governments was replaced by the governments put in place by the colonizing country and attempts to discourage the existing culture and way of life were the norm.

3 THE INTERNET

The colonization we are focusing-on in this article is the colonization of what was supposed to be a ‘free’ internet. Using the philosophy of free internet a handful of big techs have not only taken over the internet, they have created their own systems to expropriate and exploit private information of people everywhere. “Free” web applications are exchanged for the recording of every action of the user to influence them including creating a need for products and services of doubtful use[3]!

According to an article in The Atlantic[9] the internet and the web was imagined in the 1930s by Paul Otlet, a Belgian bibliographer and entrepreneur. He sketched a plan of global telescope to enable global sharing of books and multimedia[10]. Others who followed simply implemented this plan though it required development of technologies and associated how-tos. In [47] the ‘invention’ of internet is not credited to one person as a number of individuals are noted for proposing some of its mechanisms including the concept of transmission of data in the form of small packets, the addressing mechanism[93] etc. Some of these had to evolve

with the connection of more and more computers. Also there were two separate networks one in UK and the other in USA.

The concept of the interconnected documents was also mentioned in a 1945 article ‘As We may Think’ in The Atlantic by Bush[92]. This was followed by hypertext documents of various types[105]. The developments at Conseil européen pour la recherche nucléaire(CERN) in the late 1980s used this basic idea and a simple application layer protocol on top of the tranpost layer protocol(TCP) and the internet protocol(IP). The hyper text transport protocol(HTTP) was yet another application similar to the existing file trasfer protocol(FTP), simple mail transport rotocol(SMTP), domain name server(DNS) to use the underlying Transmission Control Protocol(TCP) to enable a hypertext connection between devices, one being a client the other a server. Also a rudimentary syntax, which has to be upgraded many times, for creating the textual documents and the links among these was proposed[53]. Many other people have contributed to original idea of hypertext and have since to the development of the web, most of these has been for exploiting and monetizing under the world wide web consortium(W3C). It is like the western myth that Columbus discovered America: he did not - he thought he had reached Asia. Incidentally america did not exist!! Hence web’s “invention” can’t/mustn’t be attributed to a single person. Rather what was done was an implementation of an application using existing ideas and the flexibility built into TCP/IP, OSI[110]². The current web is as different as today’s airplanes from the one imagined by Icarus and Daedalus.

As noted in [23] “even before the introduction of the web, the internet had made it possible for people to communicate via electronic mail (email)[52] and on-line chat (talk), allowing sharing of files using anonymous file transfer protocol(FTP), news(Usenet News), remote access of computer (telnet) Gopher(a tool for accessing internet resources), Archie (a search engine for openly accessible internet files) and Veronica (search for gopher sites). These early systems afforded the opportunity of interconnecting people (who wanted to be connected), sharing resources without requiring anything in return and providing security and privacy; there was not yet any question of monetizing; the whole concept was to share and there was no attempt to exploit! However, these systems were not adopted widely: the problem with these internet tools was the need to have computing savvy; the other problem was the lack of an infrastructure to transfer the know-howto. Incidentally this was also the requirement for the early web with the use of a user unfriendly, text-based web browser and lack of training facility and easy to learn tools to build and maintain hypertext documents. Some early attempts to create software for hypertext[108] were buried by the emergence of the early form of the tech giants who were more interested in having their system be the internet and crippling the users from learning the basics.

²In April of 1993, CERN put the web software in the public domain; in June of 2021 a non-fungible token (NFT) of world wide web source code sold for \$5.4m - another monetizing ploy!

4 INTERNET COLONIZATION

The author had chaired/co-chaired a number of workshops during the early meetings of WWW[14, 15]. A system at a clients site called WebJournal was under construction in mid 1994[26]: it was to provide a record of web sites discovered during a web journey and thus provide a record for latter reference[27]. This information was recorded locally and there was no need for searches. However, the option of using a web robot to scour the internet and to create a comprehensive list of web pages and index them was introduced later. The concept of robots as well as many other features used to track users was not part of the initial design from CERN. These were introduced by W3C, dominated by USAian business, to serve the needs for tracking among other monetizing tools.

Altavista, one of the early search engines, was introduced by Digital Equipment Corp.(DEC) in December 1995[106]; it had a simple design but due to many management blunders lost the search war and was shutdown in 2013. An upstart searching application, Google, claimed to be a better search engine because their search result rankings were based on the number of 'respected' pointers pointing to the page. Even though Google, whose results in the beginning were middling as shown in the tests reported in [18, 22], soon took over the lead and now has the playing field to itself. Its sheer global coverage and complete control of the digital publicity marketing, including the publicity trading exchange, the main buyer and seller[41], has prevented local search engines from emerging and challenging this dominance[23].

Over the last few years, another USAian search engine that promises not to track users called DuckDuckGo has had some success. CLIQZ was a recent example of an European attempt to create a more open search engine integrated with a web browser[34]: on their web site they point out the colonization issue: "Europe has failed to build its own digital infrastructure. US companies such as Google have thus been able to secure supremacy. They ruthlessly exploit our data. They skim off all profits. They impose their rules on us. In order not to become completely dependent and end up as a digital colony, we Europeans must now build our own independent infrastructure as the foundation for a sovereign future. A future in which our values apply. A future in which Europe receives a fair share of the added value. A future in which we all have sovereign control over our data and our digital lives. And this is exactly why we at Cliqz are developing digital key technologies made in Germany." [34]

Alas, on April 29, 2020[33] Cliqz story was over. According to the team, they were able to build an index from scratch and introduced many innovations but combined with the Covid-19 pandemic and the continued dominance of the other systems, realized that there is no future for Cliqz. The CLIQZ team built a browser that protected users' privacy using a powerful anti-tracking and content blocking technology. And of course a search engine. Yet they did it with a modest budget and attracted top talent.

Even though CLIQZ had daily users in the hundreds of thousands, they were not able to meet the cost due to the

inertia of users continuing to favour the colonizing giants. Worst of all the political stakeholders, both in Germany, where CLIQZ was based and the EU, have not woken up to the fact that they are supporting a colonized Internet and the colonizing power is USAian big tech with tacit support of successive USAian governments. After a heroic attempt, the search engine has hit the dust(cloud?), the CLIQZ browser is still around - at least for a while. Currently, in most countries of the world, even though they may have a local search engine, the lion's share of the search is using Google!

It is evident that most democratic countries need their own digital locally regulated internet infrastructure. The myth that the Internet is free and open has been exploited by many. The world deserves a fairer democratic non-colonized Internet, web and online social networks(OSN).

According to [68] "Politicians and public officials were complicit in Facebook's legitimization as a political forum. Special credit has to go to Barack Obama, as a presidential candidate in 2008, for demonstrating that a power base located on Facebook could take you all the way to the White House." This platform has been used by other politicians, dictators and political parties, and others to swing elections and mold peoples perception of reality and present an 'alternate' reality which is usually a mirage at best and in reality an untruth. "But the company has repeatedly failed to take timely action when presented with evidence of rampant manipulation and abuse of its tools by political leaders around the world"[50, 51, 55].

The large internet companies, using the advantage of the early start, the protection of the USAian government and the guise of net freedom have been enjoying a non-level playing field in web technology. The presence of a colossal corporation in search and on-line social networks is preventing any other attempts to succeed. Every time the issue of regulating big-tech comes up in the USA, the big-techs and their allies start fear mongering about giving China the advantage. This distracts from the important question of addressing the issue of exploiting the users' data and violating their privacy; take away the range of choices by offering a limited number of options beneficial to the big-techs[90]. The OSNs have killed the early attempts to create software for establishing one's own web presence, not only for individuals but also for most small organizations. One of the first things these OSNs did was to recruit USAian politicians and showed them the ease with which they can, with very little computing savvy, set up their interactive web presence and share it with their electors. Others followed and they all joined in like lemmings. In a way the OSNs have become a road-block for personal and community based sharing systems with local control.

Another problem has been the lack of imagination and inaction of most western governments, postal services and telecom utilities to provide the tools. The governments have a false faith in a free market which has never been free: the big ones dominating any start-ups and competition. These giants have become too large to regulate, and they have a large network of lobbyists and lawyers with direct access to the legislative and executive bodies of the USAian government.

There is not a single international search engine of any size that is not headquartered in the USA. The attempt by Cliqz failed in spite of their success in creating their own system and integrating it in a privacy preserving browser. Another example of this imperialist push, as noted in [23] was the attempt by Facebook to have a completely controlled free service provided by Indian telecom carrier which would have Facebook as the center with a few other services chosen by Facebook. This was an attempt by Facebook to make itself the Internet for potentially over a billion users on the sub-continent.

Most governments have not come around to adequately tax these foreign companies. Even the recent attempt by the G7 countries to impose a paltry 15% income tax[7, 70] has loopholes and most of these big-techs hardly pay a tax of even 4%[70]. One wonders why the governments do not tax these companies on the revenues earned in the country, regardless of the location of the big-tech. It is so easy with today's database and data analytic tools to determine the revenue earned in each country and tax the companies on this revenue and not allow them to play the shell game. However, incompetent politicians would not listen to even their own civil servants much less ask them to implement the system and put in laws. Of course these laws would have to override any 'free' trade agreement or even walk out from them.

A limited number of colonizing on-line social networks have attracted people from all parts of the world and given the despots around the world the ability to be heard everywhere without any checks and balances. The other issue is that some governments are trying to control such networks who have to comply so as not to lose their income from the country. Case in point is the recent attempt in India to remove contents critical of the government. China is forcing Apple to host all data of their citizens in China. In this case this data would likely be accessible to the communist government: Apple has no choice but to comply since it would risk a large portion of its global business in China and most of their manufacturing facilities. In this way, Apple has become an instrument to present a government-controlled version of the internet[58].

5 CLOSED SYSTEMS

The marketing of computing systems in the early days included the bundling of basic software support. This included the operating system, the compilers and libraries as well as training manuals. An organization would either buy the bundle or lease it and develop the specific software applications for its own use in-house. Computer Science evolved to train people who would develop this application software. The competitors to IBM, the most successful manufacturer of bundled systems, were software only houses. These competitors, using the courts and USA's anti-trust laws, were successful in un-bundling software from the hardware. The anti-trust case was based on the rationale that people who wanted software should not have to buy the hardware as well. This anti-trust case was finally dropped but it gave rise to a number of software houses. This and the idea of one

size fits all led to the establishment of software houses which produced sets of generic software that could be used for many businesses and replace the in-house systems. The concept of a bundled system that IBM used in the mid-sixties for its System 360 was subjected to litigation and prompted the unbundling of software and hardware. This un-bundling and the introduction of the PC by IBM which was an un-bundled hardware system gave rise to the birth of one of the five big-techs: sometimes called the fearsome five[65].

What has happened now is closed devices are sold today that have the software, including tracking sub-systems, built into them[45]. All the software applications(apps) created by independent software houses are installed via the operating system of the device and the device maker imposes a percent of any revenues earned by the application. There is no move anywhere to unbundle software, including the applications and the hardware. This is clearly against the spirit of the System 360 settlement. However, it has been, to date successfully used by the big-techs and is being imitated by others. One would expect that since such closed-device makers are controlling the 'application store' they would have some diligence in ensuring the quality of the software they make available and take a hefty percent of the revenue earned by the application maker. Recent articles in the press have shown that some of this software, as usual has bugs and security loopholes which could be hacked by spyware makers. One of these is attributed to a spyware firm in Israel which has targeted activists[77]. These systems, especially Internet of Things (IoTs, cell phones being one of them!), lock in the users data without providing a method for user to take care of their own data. An architectural solution and proof of concept to address this problem was presented in [1, 20].

The latest trend is abandoning in-house systems including email systems by moving them to a cloud run by one of these tech giants. The promise of tremendous cost savings is often an illusion. From the experience reported in a report by the Canadian Senate and the Wikipedia page[82, 91], one can see the fiasco caused by the system "Phoenix", mentioned earlier, bought by the Harper government to save money. After five years of continued complaints about underpayments, overpayments, and non-payments, due to a software system that was supposed to save \$70 million a year to fix Phoenix's problems, it will cost Canadian taxpayers up to \$2.2 billion by 2023 according to a Senate report.

The truth of the matter is that these big-techs are too big and have colonized the internet. The big-tech business model is to get as many people as possible to spend as many hours as possible on its site or their device so that they can sell those people's attention to advertisers. The myth that the internet is free is a farce. Each society, each city, each community must have their own contents under their own jurisdiction and control of accountable elected officers.

Some of the for-profit big-techs that run social media, make a claim that they support social justice; however, their products and their marketing models do not reflect this lip-service[49]. They claim that they spend billions of dollars

for work on AI to address these problems, but their model uses the research which shows that divisive contents attract and keep the audience. Also, how much of these billions is to support tools to weed out objectionable material including hate speech, pedophilia and false claims[83]. Some administrations have been known for promoting “divide and conquer” practised by invaders over centuries. By inventing a tag such as “newsworthy” for any content that violates accepted decorum but coming from some political figures is allowed because such contents are judged, not by independent observers but the big-techs themselves, to be ‘newsworthy’. The label does not consider inaccuracies or falsehoods nor whether it is hateful[28, 66, 89].

6 FALLOUT FROM THE COLONIZATION

The big techs have convinced billions of users that the service they are getting is free. If we ignore the intangible cost in terms of the loss of privacy and hawking of personal data, images, opinion etc. to anyone who is willing to pay for them, the service offered by these big-techs is really NOT free! In order to access their service or any other, there is the cost of device and bandwidth needed to access these services. The device will cost from several hundred dollars to a couple of thousands. The communication costs, ranges from 50 dollars to up to 100 per month. So the consumer is paying. In addition to these costly “free” services, businesses such as utilities, credit card companies and others want their customers to be billed electronically. This would save them the mailing costs but they do not discount the users bill by a corresponding amount. There are no regulations about passing such costs back to the customer and one wonders if there ever would be! One can set up a personal mail service for a few dollars a month. As noted in [44] when a product is free, the user is the product.”

These companies are invading new territories where there are no regulations. Another such plan hatched is Amazon’s Sidewalk[3]³: Amazon is one of these big-tech businesses and Sidewalk is a bridge-scheme they have hatched to ‘steal’ a users bandwidth with no remuneration and no guarantee that it could not be used to hack into a person’s system! Amazon Sidewalk works by creating a piggy low-bandwidth network using someone’s smart home devices the person has purchased; it also uses the persons telecommunication bandwidth without any permission except an opt-out. The system would likely be extended to devices and applications from third-parties that Amazon would later license. Since there are no regulations, this company is basically stealing the bandwidth, however small, from customers who are naive enough to pay for an untried product that they may not really need just because, as usual, they are hyped up and marketed to target a persons insecurity, and concern for safety and security.

³Running out of monikers - this word reminds one of the fiasco Google made with its version of Sidewalk in Toronto waterfront project and withdrew when they did not get their slice of the pie[2, 32, 40].

Since there are no regulations and laws to control the internet and its components such encroachment on privacy and security will continue. The internet including the mobile phones, have become a gold rush of our times and anyone, with support from venture capital, can stake a claim. Systems are designed as closed systems and they allow any third-party and applications to access the users data. The rush is to grab all possible types of user data, and this includes financial institutes which charge a fee for a customers account. Recently, RBC, the largest bank in Canada made a condition of on-line banking for its customers to give permission to use, anonymized data of their on-line transactions in any way they see fit to any third party they choose without concern for thier customer privacy⁴. As usual, the on-line form for giving this permission was innocuous but when one follows the link there is a 37 page document with all the legalese. One wonders where the regulating agency is and what are they regulating? If lawsuits and litigation, mostly based on monopoly legislature, are the only way[38], the entire system is going to be bogged down for years to come and may not be satisfactorily resolved. A lawsuit against Facebook, launched in British Columbia, Canada is going back and forth from one court to the another since 2013[30, 59].

World · Editor's Note

Why CBC is turning off Facebook comments on news posts for a month



Social media attacks on our journalists and the subjects of our stories is something we take seriously



Brodie Fenlon · CBC News · Posted: Jun 15, 2021 4:00 AM ET | Last Updated: 10 minutes ago

Not practicing what you preach!



WE'VE BEEN CLOSING BANKS

We Need a Few Minutes of Your Time

Please take a few minutes to review the terms of the Electronic Access Agreement. You'll need to accept the terms to continue using RBC Online Banking and the RBC Mobile app¹.

What does the agreement cover?

The Electronic Access Agreement covers your use of Online Banking and Mobile Banking, as well as other digital and third-party services you may use, like Interac[®] e-Transfer transactions.

Anything in particular I should know?

Yes. Depending on the type of account you have, any statements, notifications, or other information you currently receive on paper will change to electronic when you accept the agreement. But don't worry — if you need to switch back to paper, we'll give you the opportunity to do that after you accept the agreement.

Read & understand 37 pages of legalese in minutes!

⁴There are ways to identify a person from anonymized data[63]

One of the reasons that these big-techs have become so large lies squarely with the media - to these one can add governmental agencies, public institutes -including universities and private businesses. They have all rushed in to get 'free' exposure. They are providing legitimacy to these robber barons of our age by strategically displaying the logos of these mammoths on their own web sites; these logos take visitors to these sites to the big-techs sites where they may require them to sign-in/log-in for interaction. This gives these big-techs new victims to mine their personal data. For example most universities have a presence on the big-tech sites because the others are there - even though they have their own web site over which they have complete control. Any way, we are in the vicious circle: agencies, organizations and universities are using these sites because more people are there; however, more people are there because all of the above are promoting and using these sites! Makes one wonder what the I in the new high office called CIO stands for!

Why can't universities, centers for education, manage their own interactions with their alumni, students and prospective students and not have to go through these third parties?⁵ Furthermore, many organizations allow/encourage users to log-in to their systems using the credential for Facebook or Google! Effectively they offer their customers as sacrificial lambs to these tech giants who push the boundary of civic decencies for a greater share of the market. The media is full of items chock full of quotes of posts and tweets; these in turn lure more unsuspecting souls to be trapped in the web of these monopolies and provide them with more tons of personal data.

6.1 Survival of Newspapers and Journalism

Another fallout of the monopolies established in email, web search social networks, cell phones, computing devices and shopping is the effect it has on journalism and local newspapers. Recently, an open letter to the Prime Minister of Canada was published in many Canadian newspapers[56] to communicate the following:

"For months, you and the Minister of Canadian Heritage, Steven Guilbeault, have promised action to rein in the predatory monopoly practices of Google and Facebook against Canadian news media. But so far, all we've gotten is talk. And with every passing week, that talk grows hollower and hollower.

As you know, the two web giants are using their control of the Internet and their highly sophisticated algorithms to divert 80% of all online advertising revenue in Canada. And they are distributing the work of professional journalists across the country without compensation.

⁵It is argued that the reason people flock to these OSN sites is because of their 'better' interface: this is a fallacy. The OSN interface may not, necessarily, be better! Most people are comfortable with what they are used to and are too reluctant (busy/lazy) to learn. Also, these interfaces have evolved.

This isn't just a Canadian problem. Google and Facebook are using their monopoly powers in the same way throughout the world - choking off journalism from the financial resources it needs to survive.

. . .

In fact, the health of our democracy depends on a vibrant and healthy media. To put it bluntly, that means that you, Prime Minister, need to keep your word: to introduce legislation to break the Google/Facebook stranglehold on news before the summer recess. It's about political will - and promised action. Your government's promise.

The fate of news media in Canada depends on it. In no small way, so too does the fate of our democracy."⁶

If one looks at the fight put up not only by Google and Facebook which amount almost to blackmail but also by the USAian government recorded in the submission[5] one understands the part this government plays in this type of colonization. It is hard to understand how these submissions fail to see that Australia was addressing the market failures with digital news content and digital advertising by combining elements of the French Press Publishers' Rights the collective bargaining of publishers' licensing against the market power of publishers, as well as a novel process for the negotiation and, if necessary, arbitration of prices[12, 75].

The USA, which has put in a 'platform' designation for many of these big-techs and thus exempted from all requirement of diligence of what appears on their site seems to be behaving like the colonial governments did in the early days of the East India Company; they made it easier for these companies to exploit people, enslave them figuratively or literally using their military power and using the divide and conquer rule. In the case of the Australian draft law to require Google and Facebook to deal with a consortium of news media it is just to provide a balance. It submitted an opposing brief as being not appropriate to put in collective bargaining by any number of media players to bargain together as not respecting the principles of competition⁷[35]. The tech giants using their size and the political connections, are able to dictate their own unfair terms to news media for the use of their content. Even the treatment of their own employees could also seem to be callous.[57]

One also notes the submission [6] which includes self praise and points to the initial design of the link and free access without any mention of the monopolization of the internet and exploitation of personal data for exorbitant private profit. One wonders what part was played by W3C[112] in the introduction of cookies and tracking and other tools not in any design since Otlet!. There is no concern about copyright, fair practice, ownership, privacy etc.: no credit to all the others who had put forward the ideas of hypertext. The reading of such submissions makes one think of the famous

⁶Alas, nothing was done and summer recess has been called. In the meantime these companies control completely the digital advertisement market[41].

⁷Are the big-techs respecting the principle of competition?

lines, attributed most likely to some prolific author uttered so often to the delight of the class, by many high school teachers to one of their unruly students: "It is better that you keep your mouth shut"

7 A POSSIBLE SOLUTION

In addition to the introduction of regulations and legislation to protect privacy and stop the trade of user data by tech companies and businesses the only way to liberate ourselves from this internet colonization is to stop using these colonizing, so called, 'free' services[46].

The teaser figure above shows the statue of the salt march which was the start of the struggle against the British colonization of India[88]. Similarly, the fight against the colonization of the internet and to stop the violation of human rights by hijacking of peoples privacy must start. This fight should involve not only ordinary users but also organizations. The latter should stop using the logos of these big-techs on their web site and thier log-in credentiala. Since most of these organization already have a web site, they should invest in infrastructure to add interaction with their users and customers. This would not only remove duplication but also stop sacrificing their users and clientele to feeding the big-techs. Since the path to a web-site is the browser, they must be more privacy consious and dis-allow tracking and fingerprinting etc. Also, they currently have a facility to store log-in credentails, this must be made more robust and open. So a user should be able to access the data which must be stored locally with a secure password - thus the user needs to remember but one passowrd.

The big-techs, specially the OSNs, have marketed that the traffic to the organization's web site hosted without charge on the OSN 'free' site would increase because of the large numbers of OSN users. However, one link is just as good as another and any user with any brains should be able to find an organization because of all the search engines including those that do not track. Furthermore, the presence of the independent organizations' pages are instrumental in increasing the number of users and traffic on the OSNs.

For the ordinary user, there is another way to get a web presence and sharing without using OSNs. Since these users need to use an internet service provider(ISP) most of them also provide a web presence and email service: users should look at these services as an alternative. As more people use these alternatives, the services would gradually improve and additonal services may be added. One can, using open source system such as Linux set up one's own server. Linux has many distributions; some of them are forthe not too technical savvy and help is provided via numerous discussion forums. Educating ordinary users in the mystrey of web and databses aetc. should be addressed. One attempt by the author, and a colleague is to publish a text and the code for a set of examples; this could allow even a novice to set up a database driven web presence; this volume is 'open access" under a copy-forward scheme[24, 25].



Do they need these OSN? They have own Web sites!

In [20], a scheme to protect the users' privacy by keeping all digital data of IoTs, including cell phones, under control of the user was introduced. It is also needed to introduce regulation and legislation to crack open the closed systems used by many of these big-techs. Democratic countries must not wait for the USAian government to start the process since USA is protecting its big-techs as evidenced by the presentations made by various USAian agencies during the Australian senate hearings on news media[5]. Any sane judge would uphold laws and regulations allowing a balance in negotiation between a giant and a consortium of local small publishers. Also, according to the legal opinion cited in[75], the most recent trade agreement between Canada, Mexico and USA, there is the "ability to take legitimate policy actions in the public interest, including with respect to health, the environment, indigenous rights, and national security; and for Canada to take measures to promote and protect its cultural industries. Action taken under the authority of the exceptions is permitted even if it otherwise would have violated obligations in the Agreement."

The newspaper publishers should also provide access to their digital contents either free or for a very small fee; many newspapers already do this: however, the cost is too high! This access, in addition to providing content, could be extended to include community discussions and forums. Also to provide the subscribers means to interact with others, set up forums to discuss local concerns, provide pointers to local resources and provide the civil network for the community. If more readers go directly to the media's web sites, they would be tempted to make a voluntary contribution to support these services. Recently, Le Presse, a French language newspaper in Montreal, went all digital and became a non-profit organization. In this way they could accept donations from readers. However, since most news media are privately

owned by for profit corporations, they would not be willing to give up the ownership. However, some means of opening up and providing more community services would be one way for them to survive. The cost of creating an application for interaction and feedback is not astronomical! Over the years, the author has assigned a group project to students in an undergraduate course in databases. The project usually requires a web based application to mimic one of the social media systems. Most of the implementations are better than the initial system developed by some drop-outs and pushed with the help of some incompetent USAian politicians and businesses around the world and groupie users.

This model could be extended to other media such as videos, and then the community could have a streaming service not controlled by giants but by a local consortium. As one notices, the giant streaming services are controlling the production of, at most mediocre, contents. The best content, the classical ones are no longer to be found. The irony of this is that the production by these giant streaming services is financed by our taxes awarded by naive politicians looking for a trickle down effect. Alas there is none since the rich management of all these giants have built dams to prevent this loss!

Acknowledgement

The author would like to acknowledge the valuable discussions with and the contribution of Drew Desai (Univ. of Ottawa) and Sheila Desai(BytePress); also, many researchers and journalists cited and perhaps missed; these have been valuable in preparing this article.

REFERENCES

- [1] Ayberk Aksoy; Bipin C Desai: *Heimdallr.1: A system design for the next generation of IoT's* ICNSER2019: March 2019 Pages 92–100 <https://doi.org/10.1145/3333581.3333590>
- [2] Andrew Clement: *Sidewalk Labs' Toronto waterfront tech hub must respect privacy, democracy*, The Toronto Star, Jan. 12, 2018, <https://www.thestar.com/opinion/contributors/2018/01/12/sidewalk-labs-toronto-waterfront-tech-hub-must-respect-privacy-democracy.html>
- [3] Alex Hern: *Amazon US customers have one week to opt out of mass wireless sharing* The Guardian, Jun 1 1, 2021 <https://www.theguardian.com/technology/2021/jun/01/amazon-us-customers-given-one-week-to-opt-out-of-mass-wireless-sharing>
- [4] Annalee Newitz: *A Better Internet Is Waiting for Us* NY Times, Nov. 30, 2019 <https://www.nytimes.com/interactive/2019/11/30/opinion/social-media-future.html>
- [5] *Submissions received by the Committee* Item 13, 13.1, 17 https://www.aph.gov.au/Parliamentary_Business/Committees/Senate/Economics/TLABNewsMedia/Submissions
- [6] *Submissions received by the Committee* Item 46 https://www.aph.gov.au/Parliamentary_Business/Committees/Senate/Economics/TLABNewsMedia/Submissions
- [7] Alan Rappeport; Liz Alderman: *Yellen Aims to Win Support for Global Tax Deal* NY Times, June 3, 2021 <https://www.nytimes.com/2021/06/02/us/politics/yellen-global-tax.html>
- [8] Adam Satariano: *Facebook Faces Two Antitrust Inquiries in Europe* NY Times, June 4, 2021 <https://www.nytimes.com/2021/06/04/business/facebook-eu-uk-antitrust.html>
- [9] Alex Wright: *The Secret History of Hypertext* The Atlantic, May 22, 2014 <https://www.theatlantic.com/technology/archive/2014/05/in-search-of-the-proto-memex/371385/>
- [10] Alex Wright: *Cataloging the World: Paul Otlet and the Birth of the Information Age* Oxford University Press 2014, ISBN: 9780199931415
- [11] Ben Brody; David McLaughlin; and Naomi Nix: *U.S. Google Antitrust Case Set to Expand With GOP States Joining* Bloomberg, September 11, 2020 <https://www.bloomberg.com/news/articles/2020-09-12/u-s-google-antitrust-case-set-to-expand-with-gop-states-joining>
- [12] *What Is Baseball Arbitration?* June, 9, 2011 <http://www.arbitration.com/articles/what-is-baseball-arbitration.aspx>
- [13] Bipin C. Desai: *The Web of Betrayals* IDEAS 2018: pp 129–140 <https://doi.org/10.1145/3216122.3216140>
- [14] Bipin C. Desai: *Navigation Issues Workshop* First World Wide Web Conference, Geneva, May 28, 1994, <http://users.encs.concordia.ca/bcdesai/web-publ/navigation-issues.html>
- [15] Bipin C. Desai, Pinkerton, Brian: *Web-wide Indexing/Semantic Header or Cover Page* Third International World Wide Web Conference, April 10, 1995, <http://users.encs.concordia.ca/bcdesai/web-publ/www3-wrkA/www3-wrkA-proc.pdf>
- [16] Bipin C. Desai: *IoT: Imminent ownership Threat* Proc. IDEAS 2017, July 2017 <https://doi.org/10.1145/3105831.3105843>
- [17] Bipin C. Desai: *Privacy in the age of information (and algorithms)* Proc. IDEAS '19, June 2019, <https://doi.org/10.1145/3331076.3331089>
- [18] Bipin C. Desai: *est: Internet Indexing Systems vs List of Known URLs* Summer 1995 <http://users.encs.concordia.ca/bcdesai/web-publ/www3-wrkA/www3-wrkA-proc.pdf>
- [19] Bipin C. Desai: *Privacy in the Age Of Information (and algorithms)* IDEAS 2019, June 2019, Athens, Greece <https://doi.org/10.475/3331076.3331089>
- [20] Bipin C. Desai: *IoT: Imminent ownership Threat* IDEAS 2017, July 2017, Bristol, UK <https://doi.org/10.475/3105831.3105843>
- [21] Bipin C. Desai: *The State of Data* IDEAS '14, July 2014, Porto, Portugal <http://dx.doi.org/10.1145/2628194.2628229>
- [22] Bipin C. Desai: *Search and Discovery on the Web* Fall 2001 <https://spectrum.library.concordia.ca/983874/1/rerevisit-2001.pdf>
- [23] Bipin C. Desai: *The Web of Betrayals* In Proc. of IDEAS 2018 ACM, New York, NY, USA, 12 pages. <https://doi.org/10.1145/3216122.3216140>
- [24] Bipin C. Desai, Arlin L Kipling: *Database Web Programming* <https://spectrum.library.concordia.ca/988529/>
- [25] Bipin C. Desai, Arlin L Kipling: *Database Web Programming* <https://spectrum.library.concordia.ca/987312/>
- [26] Bipin C. Desai: *WebJournal: Visualization of WebJourney* August, 1994 <https://spectrum.library.concordia.ca/988478/>
- [27] Bipin C. Desai; Stan Swiercz: *WebJournal: Visualization of a Web Journey* ADL'95 Forum, May 15-17, 1995. Lecture notes in computer science (1082). pp. 63-80. <https://www.springer.com/gp/book/9783540614104>
- [28] Kellen Browning: *Twitch Suspends Trump's Channel for 'Hateful Conduct'* June 29, 2020, NY Times, <https://www.nytimes.com/2020/06/29/technology/twitch-trump.html>
- [29] Brian X. Chen: *Buyers of Amazon Devices Are Guinea Pigs. That's a Problem* NY Times, June 16, 2021 <https://www.nytimes.com/2021/06/16/technology/personaltech/buyers-of-amazon-devices-are-guinea-pigs-thats-a-problem.html>
- [30] *Facebook class action lawsuit launched by Vancouver woman* CBC News · May 30, 2014 <http://www.cbc.ca/news/canada/british-columbia/facebook-class-action-lawsuit-launched-by-vancouver-woman-1.2660461>
- [31] Chris Hauk: *Browser Fingerprinting: What Is It and What Should You Do About It?* Pixel Privacy, May 19, 2021 <https://pixelprivacy.com/resources/browser-fingerprinting/>
- [32] Cecco, Leyland: *Google affiliate Sidewalk Labs abruptly abandons Toronto smart city project* The Guardian, 7 May 2020, <https://www.theguardian.com/technology/2020/may/07/google-sidewalk-labs-toronto-smart-city-abandoned>
- [33] CLIQZ: *Clizq story is over* April 29, 2020 <https://cliqz.com/announcement.html>

- [34] CLIQZ: *An independent alternative for the digital era* <https://cliqz.com/en/home>
- [35] Calla Wahlquist: *US attacks Australia's 'extraordinary' plan to make Google and Facebook pay for news* The Guardian, 18 Jan 2021 <https://www.theguardian.com/media/2021/jan/19/us-attacks-australias-extraordinary-plan-to-make-google-and-facebook-pay-for-news>
- [36] Damien Cave: *A An Australia With No Google? The Bitter Fight Behind a Drastic Threat* NY Times, Jan. 22, 2021 <https://www.nytimes.com/2021/01/22/business/australia-google-facebook-news-media.html>
- [37] Deb Haaland: *A My grandparents were stolen from their families as children. We must learn about this history* Th Washington Post, June 11, 2021 <https://www.washingtonpost.com/opinions/2021/06/11/deb-haaland-indigenous-boarding-schools/>
- [38] David McCabe: *Big Tech's Next Big Problem Could Come From People Like 'Mr. Sweepy'* NY Times, Feb. 16, 2021 <https://www.nytimes.com/2021/02/16/technology/google-facebook-private-antitrust.html>
- [39] David MacDonald: *Canada's hypocrisy: Recognizing genocide except its own against Indigenous peoples* The CONversation <https://theconversation.com/canadas-hypocrisy-recognizing-genocide-except-its-own-against-indigenous-peoples-162128>
- [40] Diamond, Stephen: *Statement From Waterfront Toronto Board Chair, May 7, 2020* <https://quaysideto.ca/wp-content/uploads/2020/05/Waterfront-Toronto-Statement-May-7-2020.pdf>
- [41] Dina Srinivasan: *Google Is Dominating This Hidden Market With No Rules* NY Times, June 21, 2021 <https://www.nytimes.com/2021/06/21/opinion/google-monopoly-regulation-antitrust.html>
- [42] Elia Zureik: *Israel's Colonial Project in Palestine: Brutal Pursuit* Routledge, Taylor & Francis, 2015, ISBN: 9780415836074N
- [43] Frank, Thomas: *The Pessimistic Style in American Politics* Harper, May 2020, <https://harpers.org/archive/2020/05/how-the-anti-populists-stopped-bernie-sanders/>
- [44] Greg Bensinger: *Google's Privacy Backpedal Shows Why It's So Hard Not to Be Evil* NY Times, June 14, 2021 <https://www.nytimes.com/2021/06/14/opinion/google-privacy-big-tech.html>
- [45] Geoffrey A. Fowler: *iTrapped: All the things Apple won't let you do with your iPhone* The Washington Post, May 27, 2021 <https://www.washingtonpost.com/technology/2021/05/27/apple-iphone-monopoly/>
- [46] Joe Guinan; Martin O'Neill: *Only bold state intervention will save us from a future owned by corporate giants* The Guardian, 6 Jul 2020, <https://www.theguardian.com/commentisfree/2020/jul/06/state-intervention-amazon-recovery-covid-19>
- [47] History.com Editors: *The Invention of the Internet* Oct 28, 2019 <https://www.history.com/topics/inventions/invention-of-the-internet>
- [48] Jessy Hempel: *What Happened to Facebook's Grand Plan to Wire the World?* Wired, 05.17.2018, <https://www.wired.com/story/what-happened-to-facebooks-grand-plan-to-wire-the-world/>
- [49] Jeff Horowitz; Deepa Setharamman: *Facebook Executives Shut Down Efforts to Make the Site Less Divisive* Wall Street Journal, May 26, 2020, <https://www.wsj.com/articles/facebook-knows-it-encourages-division-top-executives-nixed-solutions-11590507499>
- [50] Hogan Libbyin; Safi, Michael: *Revealed: Facebook hate speech exploded in Myanmar during Rohingya crisis* The Guardian, April 3, 2018, <https://www.theguardian.com/world/2018/apr/03/revealed-facebook-hate-speech-exploded-in-myanmar-during-rohingya-crisis>
- [51] Harari Yuval Noah: *Why Technology Favors Tyranny* The Atlantic, October 2018, <https://www.theatlantic.com/magazine/archive/2018/10/yuval-noah-harari-technology-tyranny/568330/568330/>
- [52] Ian Peter: *The history of email* Accessed 2020 [http://www.nethistory.info/History of the Internet/email.html](http://www.nethistory.info/History%20of%20the%20Internet/email.html)
- [53] Ian Peter: *History of the World Wide Web* Accessed 2020 [http://www.nethistory.info/History of the Internet/web.html](http://www.nethistory.info/History%20of%20the%20Internet/web.html)
- [54] John Barber: *Canada's indigenous schools policy was 'cultural genocide', says report* JThe Guardian, une 2, 2015 <https://www.theguardian.com/world/2015/jun/02/canada-indigenous-schools-cultural-genocide-report>
- [55] Julia Carrie Wong: *Revealed: the Facebook loophole that lets world leaders deceive and harass their citizens* The Guardian, Apr. 12, 2021 <https://www.theguardian.com/technology/2021/apr/12/facebook-loophole-state-backed-manipulation>
- [56] Jamie Irving: *Open letter to Prime Minister Justin Trudeau* June 9, 2021 <https://www.levellingthedigitalplayingfield.ca/open-letter.html>
- [57] Jodi Kantor; Karen Weise; Grace Ashford: *The Amazon That Customers Don't See* NY Times, June 15, 2021 <https://www.nytimes.com/interactive/2021/06/15/us/amazon-workers.html>
- [58] Jack Nicas; Raymond Zhong; Daisuke Wakabayashi: *Censorship, Surveillance and Profits: A Hard Bargain for Apple in China* NY times, May 17, 2021 <https://www.nytimes.com/2021/05/17/technology/apple-china-censorship-data.html>
- [59] Jason Proctor: *B.C. Appeal Court clears way for Facebook class action* CBC News May 11, 2018 <https://www.cbc.ca/news/canada/british-columbia/facebook-sponsored-stories-appeal-courts-1.4659350>
- [60] Kenan Malik: *Tell me how you'll use my medical data. Only then might I sign up* The Guardian 6 Jun 2021 <https://www.theguardian.com/commentisfree/2021/jun/06/tell-me-how-youll-use-my-medical-data-then-i-might-sign-up>
- [61] Linden A. Mander *Azis Rule in Occupied Europe: Laws of Occupation, Analysis of Government, Proposals for Redress* The American Historical Review, Volume 51, Issue 1, October 1945, Pages 117–120 <https://doi.org/10.1086/ahr/51.1.11>
- [62] The Nobel Prize *The Nobel Peace Prize 1957* <https://www.nobelprize.org/prizes/peace/1957/summary/>
- [63] Luc Rocher; Julien M. Hendrickx; Yves-Alexandre de Montjoye: *Estimating the success of re-identifications in incomplete datasets using generative models* Nat Commun 10, 3069 (2019). <https://doi.org/10.1038/s41467-019-10933-3>
- [64] McKibben, Bill: *What Facebook and the Oil Industry Have in Common* The New Yorker, July 1, 2020, <https://www.newyorker.com/news/annals-of-a-warming-planet/what-facebook-and-the-oil-industry-have-in-common>
- [65] Farhad Manjoo: *Tech's Frightful Five: They've Got Us* NY Times, May 10, 2017, <https://www.nytimes.com/2017/05/10/technology/techs-frightful-five-theyve-got-us.html>
- [66] Paul Mozur: *A Genocide Incited on Facebook, With Posts From Myanmar's Military* NYTimes, Oct 2018, <https://www.nytimes.com/2018/10/15/technology/myanmar-facebook-genocide.html>
- [67] Mali Ilse Paquin: *Canada confronts its dark history of abuse in residential schools* The Guardian, June 6, 2015 <https://www.theguardian.com/world/2015/jun/06/canada-dark-of-history-residential-schools>
- [68] Micah L. Sifry: *Escape From Facebookistan* The New Republic, May 21, 2018 <https://newrepublic.com/article/148281/escape-facebookistan-public-sphere>
- [69] Michael Sfard: *Why Israeli progressives have started to talk about 'apartheid'* The Guardian, 3 Jun 2021 <https://www.theguardian.com/commentisfree/2021/jun/03/israeli-apartheid-israel-jewish-supremacy-occupied-territories>
- [70] Nadia Calviño; Daniele Franco; Bruno Le Maire; Olaf Scholz: *A global agreement on corporate tax is in sight – let's make sure it happens* The Guardian, June 4, 2021 <https://www.theguardian.com/commentisfree/2021/jun/04/g7-corporate-tax-is-in-sight-finance-ministers>
- [71] National Centre for Truth and Reconciliation, 2015 *What We Have Learned, Principles of Truth and Reconciliation* ISBN 978-0-660-02073-0 <https://nctr.ca/records/reports/>
- [72] National Centre for Truth and Reconciliation *Canada's Residential Schools: The History: Part 1 Origins to 1939* <https://nctr.ca/records/reports/>
- [73] National Centre for Truth and Reconciliation *W Canada's Residential Schools: The History, Part 2 1939 to 2000* <https://nctr.ca/records/reports/>
- [74] National Inquiry into Missing and Murdered Indigenous Women and Girls *WReclaiming Power and Place: The Final Report* June 3, 2019 <https://www.mmiwg-ffada.ca/final-report/>

- [75] NEWS MEDIA CANADA *Levelling the Digital Playing Field* September 2020 <https://www.levellingthedigitalplayingfield.ca>
- [76] Natasha Singer: *Google Promises Privacy With Virus App but Can Still Collect Location Data* NY Times, July 20, 2020, <https://www.nytimes.com/2020/07/20/technology/google-covid-tracker-app.html>
- [77] NSO Group/Q Cyber Technologies *Over One Hundred New Abuse Cases* <https://citizenlab.ca/2019/10/nso-q-cyber-technologies-100-new-abuse-cases/>
- [78] IBM *The birth of IBM PC* https://www.ibm.com/ibm/history/exhibits/pc25/pc25_birth.html
- [79] Nicole Perlroth: *WhatsApp Says Israeli Firm Used Its App in Spy Program* Oct. 29, 2019, <https://www.nytimes.com/2019/10/29/technology/whatsapp-nso-lawsuit.html>
- [80] Phillip Inman and Michael Savage *Rishi Sunak announces 'historic agreement' by G7 on tax reform* The Observer June 5, 2021 <https://www.theguardian.com/world/2021/jun/05/rishi-sunak-announces-historic-agreement-by-g7-on-tax-reform>
- [81] Raymond B. Blake, John Donaldson Whyte *Pierre Trudeau's failures on Indigenous rights tarnish his legacy* The CONversation <https://theconversation.com/pierre-trudeaus-failures-on-indigenous-rights-tarnish-his-legacy-162167>
- [82] Robert N. Charette *Canadian Government's Phoenix Pay System an "Incomprehensible Failure": That's the nicest thing that could be said for a debacle of the first rank* IEEE Spectrum, 05 Jun 2018 <https://spectrum.ieee.org/riskfactor/computing/software/canadian-governments-phoenix-pay-system-an-incomprehensible-failure>
- [83] Kevin Roose: *Social Media Giants Support Racial Justice. Their Products Undermine It* NY Times, 19, June 2020, <https://www.nytimes.com/2020/06/19/technology/facebook-youtube-twitter-black-lives-matter.html>
- [84] CBC *Facebook 'sponsored stories' case will be heard by Supreme Court* The Canadian Press. Mar 11, 2016 <https://www.cbc.ca/news/technology/scc-facebook-sponsored-stories-1.2660603>
- [85] Sheera Frenkel; Mike Isaac: *India and Israel Inflame Facebook's Fights With Its Own Employees* NY Times, June 3, 2021 <https://www.nytimes.com/2021/06/03/technology/india-israel-facebook-employees.html>
- [86] Sam Levin: *Is Facebook a publisher? In public it says no, but in court it says yes* The Guardian, Jul 3, 2018 <https://www.theguardian.com/technology/2018/jul/02/facebook-mark-zuckerberg-platform-publisher-lawsuit>
- [87] Sarah Left: *Email timeline* <https://www.theguardian.com/technology/2002/mar/13/internetnews>
- [88] History.com Editors *Salt Match* Jan 16, 2020 <https://www.history.com/topics/india/salt-march>
- [89] Shira Ovide: *Conviction in the Philippines Reveals Facebook's Dangers* NY Times, June 16, 2020, <https://www.nytimes.com/2020/06/16/technology/facebook-philippines.html>
- [90] Shira Ovide: *China Isn't the Issue. Big Tech Is* NY Times, June 17, 2021 <https://www.nytimes.com/2021/06/17/technology/china-big-tech.html>
- [91] Senate Report: *The Phoenix Pay Problem: Working Towards a Solution (PDF)*. Standing Senate Committee on National Finance Report of the Standing Senate Committee on National Finance. Ottawa, Ontario. July 31, 2018. p. 34, https://sencanada.ca/content/sen/committee/421/NFFN/Reports/NFFN_Phoenix_Report_32-WEB.e.pdf
- [92] Vannevar Bush: *As we may think* The Atlantic, July 1945 <https://www.theatlantic.com/magazine/archive/1945/07/as-we-may-think/303881/>
- [93] G. Vinton; Robert Cerf; E. Kahn: *A Protocol for Packet Network Intercommunication* IEEE Trans on Comms, Vol Com-22, No 5 May 1974 <https://www.academia.edu/8157148/A-Protocol-for-Packet-Network-Intercommunication>
- [94] Vega, Nicolas *New York Times pulls out of Apple News partnership* NY Post, June 29, 2020. <https://nypost.com/2020/06/29/new-york-times-pulls-out-of-apple-news-partnership/>
- [95] Wikipedia *Videotelephony* <https://en.wikipedia.org/wiki/Videotelephony>
- [96] Wikipedia *Austrian East India Company* https://en.wikipedia.org/wiki/Austrian_East_India_Company
- [97] Wikipedia *Colonization* <https://en.wikipedia.org/wiki/Colonization>
- [98] Wikipedia *East India Company* https://en.wikipedia.org/wiki/East_India_Company
- [99] Wikipedia *Dutch East India Company* https://en.wikipedia.org/wiki/Dutch_East_India_Company
- [100] Wikipedia *French East India Company* https://en.wikipedia.org/wiki/French_East_India_Company
- [101] Wikipedia *Portuguese East India Company* https://en.wikipedia.org/wiki/Portuguese_East_India_Company
- [102] Wikipedia *Swedish East India Company* https://en.wikipedia.org/wiki/Swedish_East_India_Company
- [103] Wikipedia *Company rule in India* https://en.wikipedia.org/wiki/Company_rule_in_India
- [104] Wikipedia *History of hypertext* https://en.wikipedia.org/wiki/History_of_hypertext
- [105] Wikipedia *Hypertext* <https://en.wikipedia.org/wiki/Hypertext>
- [106] Wikipedia *AltaVista* <https://en.wikipedia.org/wiki/AltaVista>
- [107] Wikipedia *X Window System* https://en.wikipedia.org/wiki/X_Window_System
- [108] Wikipedia *SoftQuad Software* https://en.wikipedia.org/wiki/SoftQuad_Software
- [109] Wikipedia *Uyghur genocide* https://en.wikipedia.org/wiki/Uyghur_genocide
- [110] Wikipedia *OSI model* https://en.wikipedia.org/wiki/OSI_model
- [111] Wikipedia *Phoenix pay system* https://en.wikipedia.org/wiki/Phoenix_pay_system
- [112] W3C *World Wide Web Consortium (W3C)* <https://www.w3.org>

Data Mining Autosomal Archaeogenetic Data to Determine Minoan Origins

Peter Z. Revesz

University of Nebraska-Lincoln
Lincoln, Nebraska, USA
revesz@cse.unl.edu

ABSTRACT

This paper presents a method for data mining archaeogenetic autosomal data. The method is applied to the widely debated topic of the origin of the Bronze Age Minoan culture that existed on the island of Crete from 5000 to 3500 years ago. The data is compared with some Neolithic and early Bronze Age samples from the nearby Cycladic islands, mainland Greece and other Neolithic sites. The method shows that a large component of the Minoan autosomal genomes has sources from the Neolithic areas of northern Greece and the rest of the Balkans and a minor component comes directly from Neolithic Anatolia and the Caucasus.

CCS CONCEPTS

• **Computing methodologies** → *Machine learning; Machine learning approaches*; • **Information Systems** → *Data mining*; • **Applied computing** → *Bioinformatics*.

KEYWORDS

Archaeogenetics, Autosomal, Data Mining, DNA, Minoan

ACM Reference Format:

Peter Z. Revesz. 2021. Data Mining Autosomal Archaeogenetic Data to Determine Minoan Origins. In *25th International Database Engineering Applications Symposium (IDEAS 2021)*, July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3472163.3472178>

1 INTRODUCTION

The rapid growth of autosomal archaeogenetic data was motivated in recent years by a huge interest in tracking ancient human migrations. Unfortunately, the development of cutting-edge genetic sequencing technologies that have facilitated this rapid growth has not been matched with a proportional development of advanced data mining methods that would bring out the most useful information from the newly available data. The goal of this paper is to introduce a new data mining method that can better answer the deeper questions about human migrations.

In this paper, we show how better data mining of archaeogenetic data can illuminate some widely debated questions about prehistory. In particular, we use the example of the island of Crete. In

Crete, there was only a minimal presence of humans during the Paleolithic and the Mesolithic periods, when people lived from fishing and hunting. Agriculture as well as the keeping of sheep, goats, cattle and pigs arrived at Crete from the Near East about 8000 years ago. This is well-documented in archaeology. The Bronze Age started in Crete around 5000 years ago with a civilization that was named the Minoan civilization by Sir Arthur Evans, the famous British archaeologist who discovered the palace of Knossos in central Crete just a few miles south from the modern city of Heraklion, Greece. Even since Arthur Evans' discovery in the early 20th century, people have wondered about where the Minoans came from. Various theories were proposed with no clear answer. Bernal [2] and Evans [10] proposed an Egyptian, Gordon [12] and Marinatos [16] some Near Eastern, Campbell-Dunn [3] Libyan, Haarmann [13] an Old European, and Gimbutas [11] an Anatolian origin of the Minoan civilization. Naturally, Crete's position as an island in the middle of the Mediterranean Sea can lead to many proposals.

One of the motivations of identifying the origins of the Minoan culture is to find the linguistically closest relatives of the Minoans. For example, if they came from Libya, then they may have spoken a Fulani language [3], if they came from the Near East, then they may have spoken a Semitic language such as Phoenician or Ugaritic [12], and if they came from the north, then they may have spoken some Pre-Indo-European language such as Basque, Etruscan or a Finno-Ugric language [21]. The Minoan culture left behind thousands of inscriptions that are considered undeciphered. The identification of related languages can be an important step towards the decipherment of the Minoan scripts.

In this situation, many people hoped that archaeogenetic data will give a definite answer to the question of Minoan origins. However, the archaeogenetic studies were not very conclusive. Moreover, they apparently contradict each other in many details. The earliest Minoan DNA study was published in 2013 by Hughey et al. [14], who examined only mitochondrial DNA (mtDNA). Based only on the mtDNA data, they made the following claim: "Our data are compatible with the hypothesis of an autochthonous development of the Minoan civilization by the descendants of the Neolithic settlers of the island." That means that they thought that the Neolithic settlers of Crete stayed in place and eventually developed their Bronze Age culture. In contrast to Hughey et al. [14], a later mtDNA-based data mining study by Revesz [23] suggested a Danube Basin and Western Black Sea origin of the Minoans.

Lazaridis et al. [15] published the first autosomal, whole-DNA Minoan data in 2017. Lazaridis et al. [15] wrote the following in their conclusion: "Minoans and Mycenaeans were genetically similar, having at least three-quarters of their ancestry from the first Neolithic farmers of Western Anatolia and the Aegean, and most of

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-8991-4/21/07...\$15.00
<https://doi.org/10.1145/3472163.3472178>

the remainder from ancient populations related to those of the Caucasus and Iran." This again suggested that the Minoan civilization was developed from the local Neolithic settlers. Recently, Clemente et al. [5] succeeded to get more archaeogenetic data from Crete and other islands of the Aegean Sea between Greece and Turkey. Their main conclusion was that the Minoans had over seventy-five percent of ancestry from European Neolithic farmers.

Clearly, the current archaeogenetic results leave a confusing picture. While the genetics researchers have added much valuable data to the genome databases in recent years, their analyses seems confusing and contradictory in some details. This suggests a need for a better data mining of the archaeogenetic data, which we will develop in this paper.

The rest of this paper is organized as follows. Section 2 describes some background to the current study and related previous results. Section 3 describes our data mining method. Section 4 presents some experimental results. Section 5 gives a discussion of the results, including a timeline of the Neolithic and Bronze Age migrations that are implied by our archaeogenetic data mining results. Finally, Section 6 gives some conclusions and directions for future work.

2 BACKGROUND AND PREVIOUS RESULTS

In this paper, we describe data mining of ancient genomes with a special interest in discovering the origins of two Bronze Age civilizations. The first is the Minoan civilization that existed on the island of Crete, which is shown at the bottom of Fig. 1, and the second is the Cycladic civilization that existed on the other islands above Crete and between present day Greece and Turkey. Currently, the *European Nucleotide Archive* (ENA) contains two Cycladic and nine Minoan autosomal genomes that were added by Clemente et al. [5] and Lazaridis et al. [15]. The Cycladic civilization is represented by the samples Kou01 and Kou03 from Koufonisia island, while the Minoan civilization is represented by the samples I0070, I0071, I0073, I0074, I9005 from Hagios Charalambos Cave, Crete, samples I9129, I9130, I9131 from Moni Odigitria, Crete, and Pta08 from Petras, Crete. These autosomal genomes will be compared with autosomal genomes from other Neolithic samples. Fig. 1 shows the location of some of the Aegean samples.

Admixture analysis is an important method in genetic testing. The basic goal of admixture analysis is to find the possible sources of a test sample *Test*. In an admixture analysis, some possible options *Ref1*, *Ref2*, ..., *Refn* are given. Essentially, an admixture analysis compares the genes of the *Test* and the corresponding genes of the possible sources. The admixture analysis tallies how many times each possible source had a corresponding gene that was closest to the gene of *Test* among all of the sources. For example, the result of an admixture analysis may be that the *Test* is composed of 50 percent of *Ref1*, 30 percent of *Ref2*, and 20 percent of *Ref3*. Sometimes, the possible source population is not just one genome sample but a small set of related samples, for example, all the samples from the Minoan site of Moni Odigitria. In this case, several corresponding genes can be considered from this group with always the best out of those being chosen. In practice, the number of possible source populations is limited to a small number, because a full admixture analysis is a computationally complex task.

Clemente et al. [5] did an admixture analysis of some Bronze Age Aegean samples from the Cycladic, Minoan and Mycenaean cultures as shown in Fig. 2. They grouped these samples according to the following time periods: Early Bronze Age (EBA), Middle Bronze Age (MBA) and the Late Bronze Age (LBA). Fig. 2 shows their main results about how their Aegean samples (*Test* column) can be explained by two hypothetical source populations (*Ref1* and *Ref2* columns). As can be seen from Fig. 2, their source populations were the Aegean samples listed as Kou01, Kou03, Mik15, Log02, Log04, Mik15, Minoan_Lasithi, Minoan_Odigitria, and Mycenaean. In addition, they considered the following source populations: some of the Aegean samples themselves, Anatolian Neolithic (Anatolia_N), Balkans Late Bronze Age (Balkans_LBA), Caucasian Hunter-Gatherer (CHG), Europe Late Neolithic and Bronze Age (Europe_LNBA), Eastern European Hunter-Gatherer (EHG), Iranian Neolithic (Iran_N), Pontic Steppe Early and Middle Bronze Age (Steppe_EMBA), Pontic Steppe Middle and Late Bronze Age (Steppe_MLBA), and Western European Hunter-Gatherer (WHG).

Clemente et al. [5] apparently did not consider the European Early Neolithic and European Middle Neolithic cultures as possible sources for the Aegean samples. Lazaridis et al. [15] also did not consider those types of samples. Revesz [23] considered European Neolithic mitochondrial DNA samples from the Danube Basin and suggested that some migration took place from the Danube Basin to Crete during the Bronze Age. Our study extends the earlier works by considering autosomal, whole DNA data and include European Early and Middle Neolithic samples too.

3 DATA MINING METHOD

We have used the G25 admixture analysis package [25]. The G25 admixture analysis package already represents over a thousand archaeogenetic samples or small sets of samples as possible options to select for sources. While the previous studies limited their possible sources to a few selected ones, we did not preselect any particular source and instead considered all available Neolithic autosomal data as potential sources. This approach can avoid the different conclusions of the previous studies that were due to preselecting a very limited number of sources for consideration, while excluding hundreds of other possible sources from consideration.

Our data mining analysis started with the following open-ended question: *Where did the Neolithic ancestors of the Minoans live?* This question was completely open-ended, because we did not restrict the set of possible sources that the G25 package could choose from. Our approach was to look as widely as possible for potential sources among all Old World Neolithic, Mesolithic and Paleolithic samples from the G25 database, which was a total of 271 potential sources. Then we concurrently tested all of them as potential sources for each of the eleven Aegean samples. For each test sample, the G25 admixture analysis system listed all the hypothetical sources with a decreasing percentage order. Two examples of the G25 output results are shown in Fig 3. On the top we see the results for the Minoan sample I0070, whose origins are 51.8 percent Greek Peloponnese Neolithic, 29.6 percent other Greek Neolithic, 11.8 percent Caucasus Lowland from Azerbaijan, 6.6 percent Caucasian Hunter-Gatherer from Georgia, and 0.2 percent Pre-Pottery

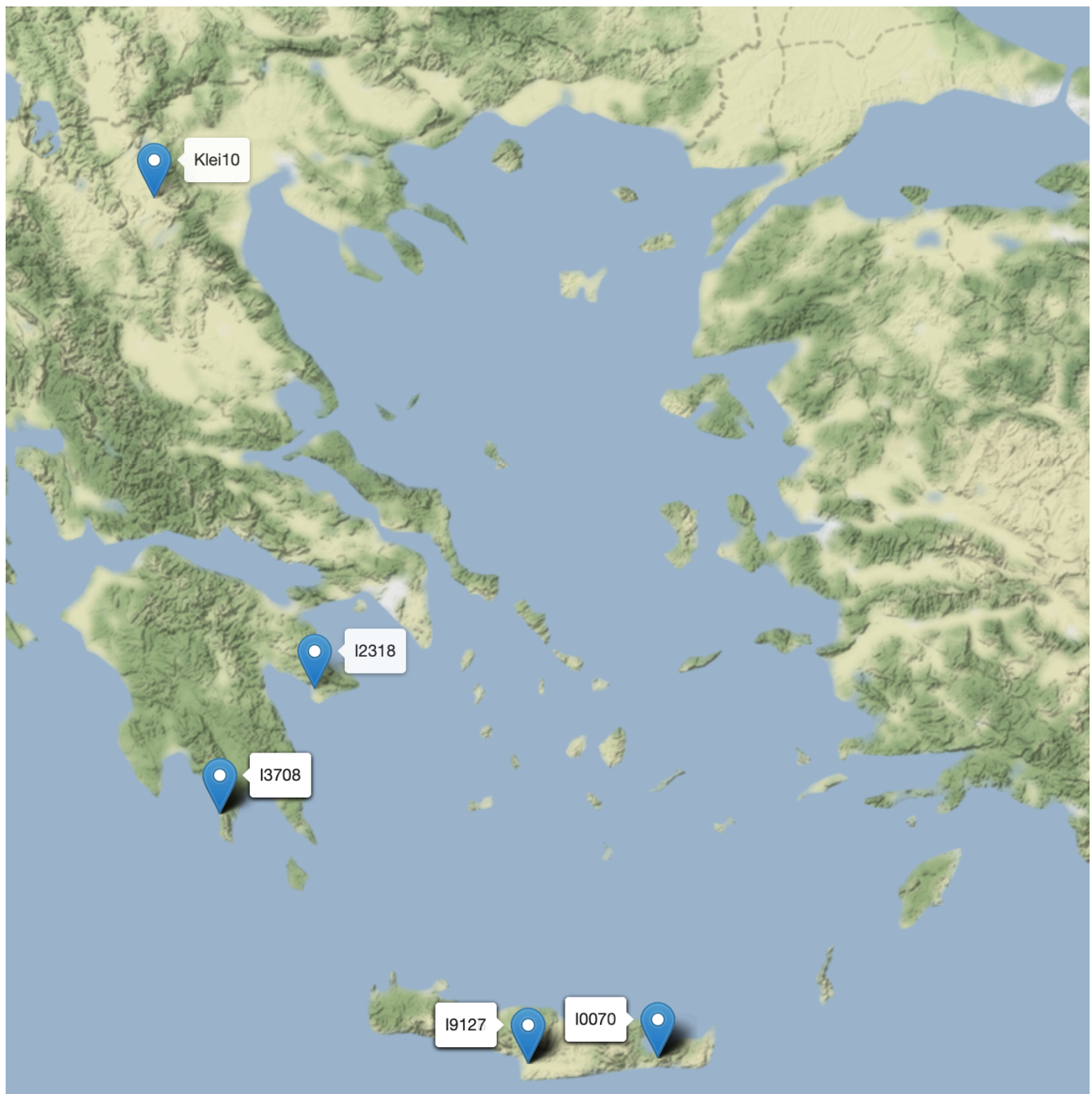


Figure 1: The location of some of the Minoan samples. On the island of Crete, shown on the bottom of the map, the sample I9127 belongs to Moni Odigitria, and the sample I0070 belongs to the Charalambos Cave. This map was generated based on the amtDB database [9]

Neolithic Culture B (PPNB) from the Levant. Below this there is another example regarding the Minoan sample from Petras identified by Pta08. Clearly, this has a very different origin. In fact, its hypothetical sources are 49.4 percent the Neolithic Starcevo Culture

from Hungary, 33.6 percent Caucasus Lowland from Azerbaijan, 7.6 percent Linear Pottery Culture (LBK) from Germany, 5.2 percent Middle Neolithic Alföld Culture from Hungary, 2.6 Neolithic

Period	Test	Ref1	Ref2	Ref3	Mixture Prop. Ref1 \pm SE	Mixture Prop. Ref2 \pm SE	Mixture Prop. Ref3 \pm SE	p value
EBA	Kou01	Anatolia_N	CHG		0.75 \pm 0.03	0.25 \pm 0.03		0.67
	Kou01	Anatolia_N	Iran_N		0.75 \pm 0.03	0.25 \pm 0.03		0.90
	Kou03	Anatolia_N	CHG		0.69 \pm 0.03	0.31 \pm 0.03		0.10
	Mik15	Anatolia_N	CHG		0.84 \pm 0.03	0.16 \pm 0.03		0.08
	Mik15	Anatolia_N	Iran_N		0.84 \pm 0.03	0.16 \pm 0.03		0.07
	Pta08	Mik15	Iran_N		0.98 \pm 0.03	0.02 \pm 0.03		0.09
	Pta08	Mik15	CHG		0.99 \pm 0.03	0.01 \pm 0.01		0.07
	Kou01	Anatolia_N	CHG	EHG	0.74 \pm 0.04	0.25 \pm 0.03	0.01 \pm 0.02	0.67
	Kou01	Anatolia_N	Iran_N	EHG	0.74 \pm 0.03	0.24 \pm 0.03	0.02 \pm 0.02	0.88
	Kou03	Anatolia_N	Iran_N	EHG	0.67 \pm 0.03	0.25 \pm 0.03	0.08 \pm 0.02	0.82
MBA	Log02	Kou01	EHG		0.81 \pm 0.02	0.19 \pm 0.02		0.07
	Log02	Kou03	WHG		0.91 \pm 0.02	0.09 \pm 0.02		0.06
	Log02	Anatolia_N	Balkans_LBA	CHG	0.22 \pm 0.05	0.65 \pm 0.06	0.12 \pm 0.04	0.08
	Log02*	Kou01	Steppe_MLBA		0.61 \pm 0.03	0.39 \pm 0.03		0.20
	Log02*	Kou01	Europe_LNBA		0.56 \pm 0.04	0.44 \pm 0.04		0.05
	Log04	Kou01	Balkans_LBA		0.21 \pm 0.06	0.79 \pm 0.06		0.08
	Log04	Kou03	Balkans_LBA		0.26 \pm 0.07	0.74 \pm 0.07		0.07
	Log04	Mik15	Balkans_LBA		0.21 \pm 0.06	0.79 \pm 0.06		0.06
	Log04	Anatolia_N	CHG	EHG	0.58 \pm 0.03	0.16 \pm 0.03	0.27 \pm 0.02	0.12
	Log04*	Anatolia_N	Steppe_EMBA		0.53 \pm 0.03	0.47 \pm 0.03		0.35
	Log04*	Anatolia_N	Steppe_MLBA		0.38 \pm 0.03	0.62 \pm 0.03		0.13
	Log04*	Pta08	Balkans_LBA		0.15 \pm 0.04	0.85 \pm 0.04		0.06
	Log04*	Pta08	Steppe_MLBA		0.44 \pm 0.03	0.56 \pm 0.03		0.36
LBA	Mycenaean	Log04	Minoan_Lasithi		0.36 \pm 0.04	0.64 \pm 0.04		0.35
	Mycenaean	Log04	Minoan_Odigitria		0.21 \pm 0.04	0.79 \pm 0.04		0.45
	Mycenaean	Anatolia_N	Kou03		0.37 \pm 0.09	0.63 \pm 0.09		0.40

Figure 2: The admixture analysis results of Clemente et al. [5] about how their Aegean samples' (Test column) can be explained by two hypothetical source populations (Ref1 and Ref2 columns).

Tepecik-Ciftlik Culture from Turkey, 1.4 percent Pre-Pottery Neolithic Culture B (PPNB) from the Levant, and 0.2 percent from Late Neolithic Malaysia.

4 EXPERIMENTAL RESULTS

From all the G25 admixture analysis system outputs, we created a table as shown in Fig. 4, where each row represents a culture that was a source of one of the eleven Cycladic or Minoan samples. We only list those rows that had some non-zero value for at least one of the eleven Cycladic and Minoan test samples. In each column, the percentages add up to 100 percent, meaning that the hypothetical sources are fully accounted.

In Fig. 4 we present the data by grouping the sources together according to regions, using a separate color for each region. African sources are shown in yellow, Greek and Macedonian sources are shown in dark blue, other European sources, which are almost all from the Danube Basin, are shown in light blue, Caucasian, Ukrainian, and Russian Siberian sources are shown in orange, while Fertile Crescent and Iranian sources are shown in green.

4.1 Detection of African Origin

The African DNA sample is from a hunter-gatherer from the Shum Laka rock shelter in Cameroon about 8000 years ago. Although the connection is only 0.6 percent for the sample 19129, it suggests that as the Sahara dried out, some people moved north into Europe, and apparently reached the island of Crete, while another group moved south to Sub-Saharan Africa. This explains some of the linguistic connections that were found between African and European mountain names [21]. This is the first time an African genetic connection was detected to the Minoans in any study.

4.2 Minoan groups at Charalambos Cave and Moni Odigitria are Distinct

Fig. 5 shows the clusters that we obtain after finding the averages for each location and then separating the low values between 0 and 34 (sky blue) and high values between 42 and 73 (red). It is now apparent that the Cycladic Koufonisi and the Minoan Charalambos Cave samples cluster together, while the Minoan Moni Odigitria and

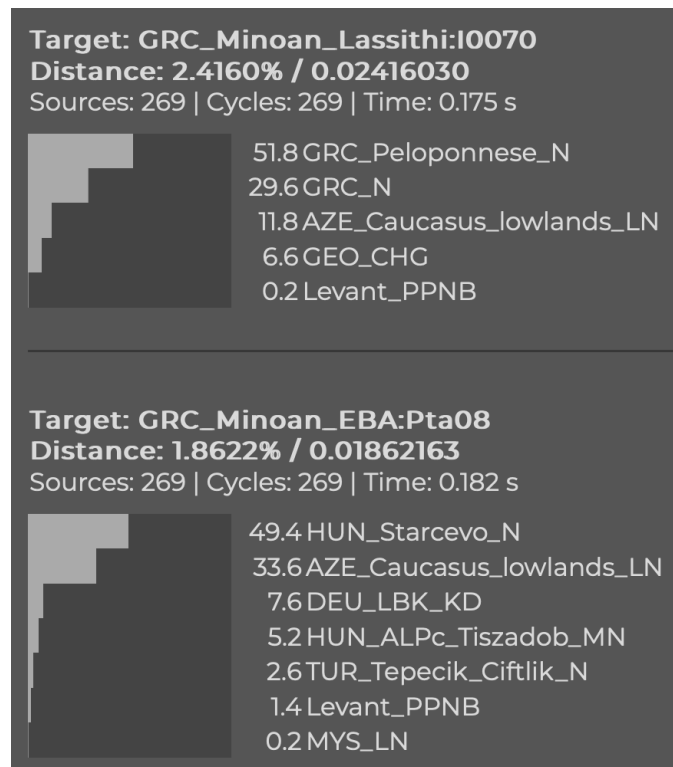


Figure 3: Example data output of the G25 admixture analysis system.

Petras samples cluster together. Clemente et al. [5] and Lazaridis et al. [15] did not note any differences between the Charalambos Cave and the Moni Odigitria groups.

4.3 Principal Component Analysis

Fig. 6 shows a *Principal Component Analysis* (PCA) of various Neolithic, Chalcolithic and Bronze Age samples from Anatolia and Southeastern Europe that are also used in Fig. 4 with the addition of a few Mycenaean samples that we added as extra for comparison.

The PCA components were automatically generated by a free PCA software tool developed for the G25 admixture analysis system and available on its github page. The first and the second principal components are mapped to the x and the y axes of Fig. 6. Clearly, the African, Caucasian, and Fertile Crescent samples are all either greater than -0.1 on the x -axis or less than -0.05 on the y -axis except some Turkish Neolithic samples, while the Danube Basin and Greek samples are the opposite, i.e., less than -0.1 on the x -axis or greater than -0.05 on the y -axis except the Corded Ware samples. Hence, the Corded Ware samples appear to have a mixed Caucasus and Danube Basin ancestry.

Fig 7 shows a detail of Fig. 6. The detail shows that the (mostly northern) Greek Neolithic samples are closest to the Hungarian Körös early Neolithic samples except the separate group of Greek Peloponnese Neolithic samples. The Charalambos Cave (marked as GRC_Lassithi_Plateau), the two Cycladic (GRC_Cycladic_EBA)

and the Petras (GRC_Minoan_EBA) samples are increasingly below and to the right of the Greek Neolithic samples. This suggests a migration from the Danube Basin to Greece with the Danube Basin signature diminishing with increasingly heavy local admixture in the Peloponnese peninsula, the Cyclades and Crete. Some of the local population may have come from Anatolia as indicated by the locations of the Anatolian Neolithic samples from Kumtepe and Tepecik_Ciftlik.

The five Charalambos Cave samples and the three Moni Odigitria samples form two different clusters without any overlap. The Moni Odigitria samples are all above the Greek Peloponnese Neolithic samples and much above the Charalambos Cave, Cycladic and Petras samples. The Moni Odigitria samples are close to several Danube Basin Middle Neolithic samples. This suggests another migration from the Danube Basin to Crete.

Finally, the Mycenaean samples are considerably to the right of the Minoan samples, suggesting that they are a mixture of Minoan and Caucasian origin.

5 DISCUSSION OF THE RESULTS

Based on radiocarbon dating the estimated age of the Moni Odigitria samples ranges from 2210 to 1600 BC, while the estimated age of the Charalambos Cave samples ranges from 2000 to 1700 BC [15]. Therefore, the Moni Odigitria samples seem older. The Petras sample is even older with an estimated date range from 2849 to 2621 BC [5]. Therefore, it is logical that the Moni Odigitria and

	Kou01	Kou03	I0070	I0071	I0073	I0074	I9005	I9129	I9130	I9131	Pta08
CMR_Shum_Laka_8000BP								0.6			
GRC_N		13.4	29.6	18	13		13				
GRC_Peloponnese_N	21.4	49.2	51.8	49.6	28.8	35.4	40.8		19.6	9	
MKD_N							31.2				
total Greece	21.4	62.6	81.4	67.6	41.8	35.4	85	0	19.6	9	0
BGR_Krepost_N				18.2		7.6				27.4	
Corded_Ware_DEU	0.6										
Corded_Ware_POL_early										1.6	
DEU_LBK_KD	1			3.6		3.2			33.4	6	7.6
DEU_LBK_SCH								2.8			
DEU_LBK_UW		4.6									
FRA_Grand_Est_EN								4.2			
HUN_ALPc_I_MN								10.6			
HUN_ALPc_Tiszaob_MN									13.8		5.2
HUN_Koros_N										45.6	
HUN_LBK_MN					32.4						
HUN_Sopot_LN									13.8		
HUN_Starcevo_N	13.4					32.2					49.4
ROU_N		10.4						47.4			
SRB_N	3.4							2.4			
SRB_Starcevo_N								7.2			
total Danube Basin	18.4	15	0	21.8	32.4	43	0	74.6	61	80.6	62.2
AZE_Caucasian_lowlands_LN	24.2		11.8	2.8	6.6	20.4	6.4				33.6
GEO_CHG		8.6	6.6	7.8	3		8.6		2.6	6.6	
RUS_AfantovaGora3		2									
RUS_Yakutia_N					0.2	0.8					
RUS_Yakutia_Ymyiakhatkh_LN						0.4					
UKR_N	14										
total Caucasus	38.2	10.6	18.4	10.6	9.8	21.6	15	0	2.6	6.6	33.6
IRN_Seh_Gabi_LN		11.8							10		
IRN_Wezmeh_N					3.8			10.2			
Levant_PPNB			0.2								1.4
Levant_PPNC										3.8	
MYS_LN											0.2
TUR_Tepecik_Ciftlik_N	0.8								6.8		2.6
TUR_Kumtepe_N	21.2				12.2			14.6			
total Fertile Crescent	22	11.8	0.2	0	16	0	0	24.8	16.8	3.8	4.2

Figure 4: The Neolithic sources for various Cycladic and Minoan autosomal DNAs according to the G25 admixture analysis system.

the Petras samples cluster together in Fig. 5. This clustering is partially supported by Fig. 7 where the Moni Odigitria and the Petras samples have the same x -axis. However, they have a significantly different Caucasus admixture. The Caucasus admixture of the Petras sample is based on AZE_Caucasian_lowlands_LN, which is located almost directly below the Petras sample. Hence it is likely that the Petras sample is lower than the Moni Odigitria samples due to the higher AZE_Caucasian_lowlands_LN admixture.

Based on our data mining, we can reconstruct the following sequence of events and illustrate them in Fig. 8.

- (1) The Neolithic agricultural revolution started in the Fertile Crescent and spread from there via three routes that are relevant to our study. The first and the second routes went slowly through Anatolia, where the spread of agriculture took two different turns. One route continued along the eastern Black Sea until the Danube Delta and then followed the Danube

	Koufonisi	Charalambos Cave	Moni Odigitria	Petras
Greece + Macedonia	42	62.24	9.53	0
Danube Basin	16.7	19.44	72.07	61.8
Caucasus	24.4	15.08	3.07	33.4
Fertile Crescent	16.9	3.24	15.13	4.6

Legend: 0 - 20 24 - 34 42 - 100

Figure 5: The cluster analysis of the four different Aegean locations.



Figure 6: A PCA analysis of the archaeogenetic samples in Figure 4.

as the gateway into Europe. The early European Farmers who followed this route established the *Old European civilization* [11]. This culture is divided into various groups such as the *Vinča culture*, the *Körös culture*, etc. The second route went along the cost of Southern Europe, reaching from Greece to the Iberian peninsula. The farmers on this second route area are known for their *Cardium pottery culture* [11]. A third route of agricultural expansion expanded toward the

Caucasus. Initially, these three groups kept separate from each other, but later there was considerable interaction and apparent convergence of material culture between these areas. Childe[4] describes archaeological similarities between the Old European civilization in the Danube Basin and the Neolithic Dimini culture in Greece. Revesz [22] traces the spread of art motifs that accompanied the early Neolithic and the Bronze Age human migrations. Revesz [22] reports

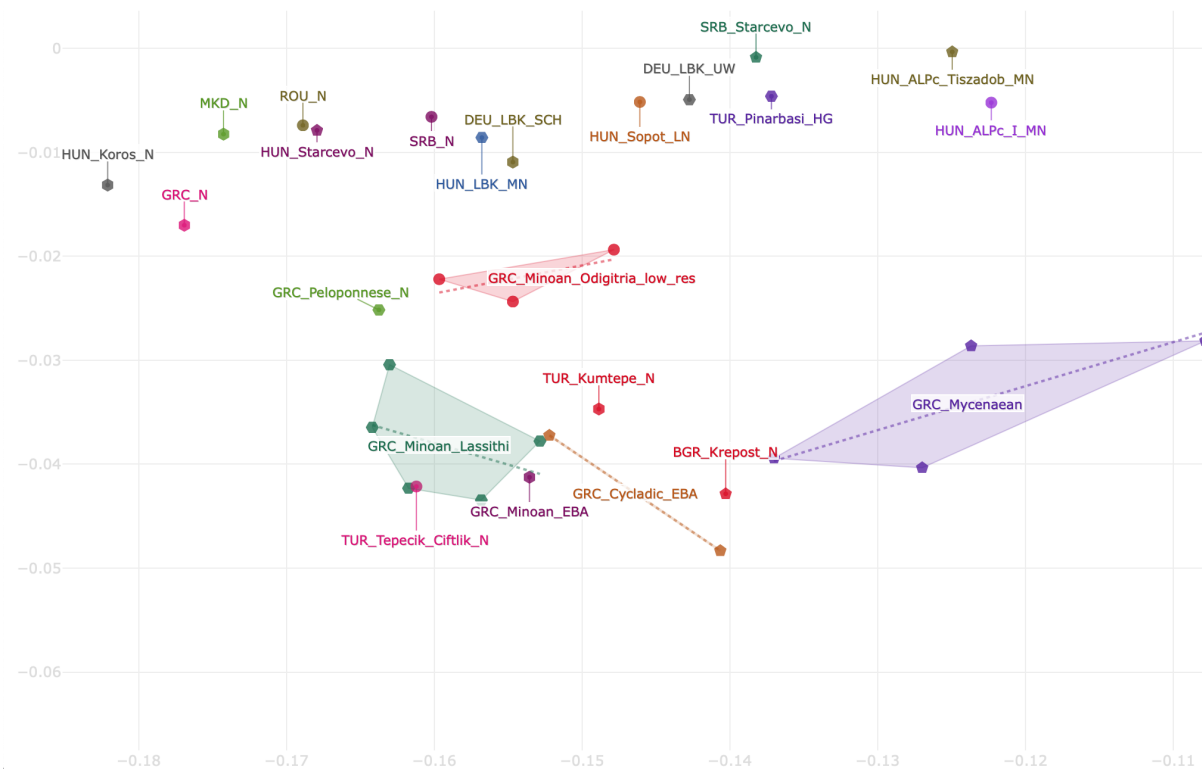


Figure 7: PCA detail with the Minoan Charalambos Cave samples, Moni Odigitria samples, and the Greek Peloponnese Neolithic samples. The dashed circles indicate the clusters of all Minoan Charalambos, all Minoan Moni Odigitria and one group of Greek Peloponnese Neolithic samples.

a particularly strong connection between the late Neolithic and early Bronze Age Danube Basin cultures and the Minoan culture.

There was also some early population movement from the Caucasus, which has effected the eastern part of Crete the most, as evidenced by the high Caucasus lowland ancestry of the Petras sample as seen in Fig. 4.

- (2) At the beginning of the Early Bronze Age, about 5000 years ago, the Minoan culture was established by people who moved from mainland Greece and Macedonia and to a lesser extent from the Caucasus lowland area to the Cyclades and to Crete as shown in Fig. 8. The EBA migration seems to have affected the Cyclades, including the island of Koufonisi, and the northern and central part of Crete, including the Charalambos Cave on the Lassithi Plateau, according to the cluster analysis in Section 4.2. The PCA analysis suggests that the Charalambos Cave samples genetically originated from the Greek mainland, especially the Peloponnese peninsula, but they mixed with people who came from the Caucasus. Concurrently, some EBA migration could have occurred to eastern Crete from the Danube Basin area, which was part of the Old European civilization [11]. This may explain the high concentration of Danube Basin ancestry in the Petras sample.

The migration to Petras could have been easily done by ship starting from the Danube Delta area and sailing through the Bosphorus Strait and the Dardanelles Strait to the Aegean Sea and then following the western coast line of Anatolia as shown in Fig. 8.

Possible Cause: According to Kurgan Hypothesis [11], the first groups of Indo-Europeans started to move from the Pontic Steppe to Central and Southeastern Europe around 5000 years ago. Hence, it seems that the Indo-European migration, and in particular the migration of Mycenaeans to the Peloponnese area caused some of the populations to move south to the Cyclades and Crete.

- (3) At the beginning of the Middle Bronze Age, about 4200 years ago, the Middle Minoan culture was created by another movement of people to Crete from the Danube Basin area. This wave of migration also followed the shipping route shown in Fig. 8 but apparently after reaching the eastern end of Crete, people sailed along the southern coast of Crete in an east-to-west direction before reaching Hagia Triada, a natural harbor near Moni Odigitria. This hypothesis is supported by the remarkably high Danube Basin admixture and no Caucasus lowland admixture in the Moni Odigitria samples

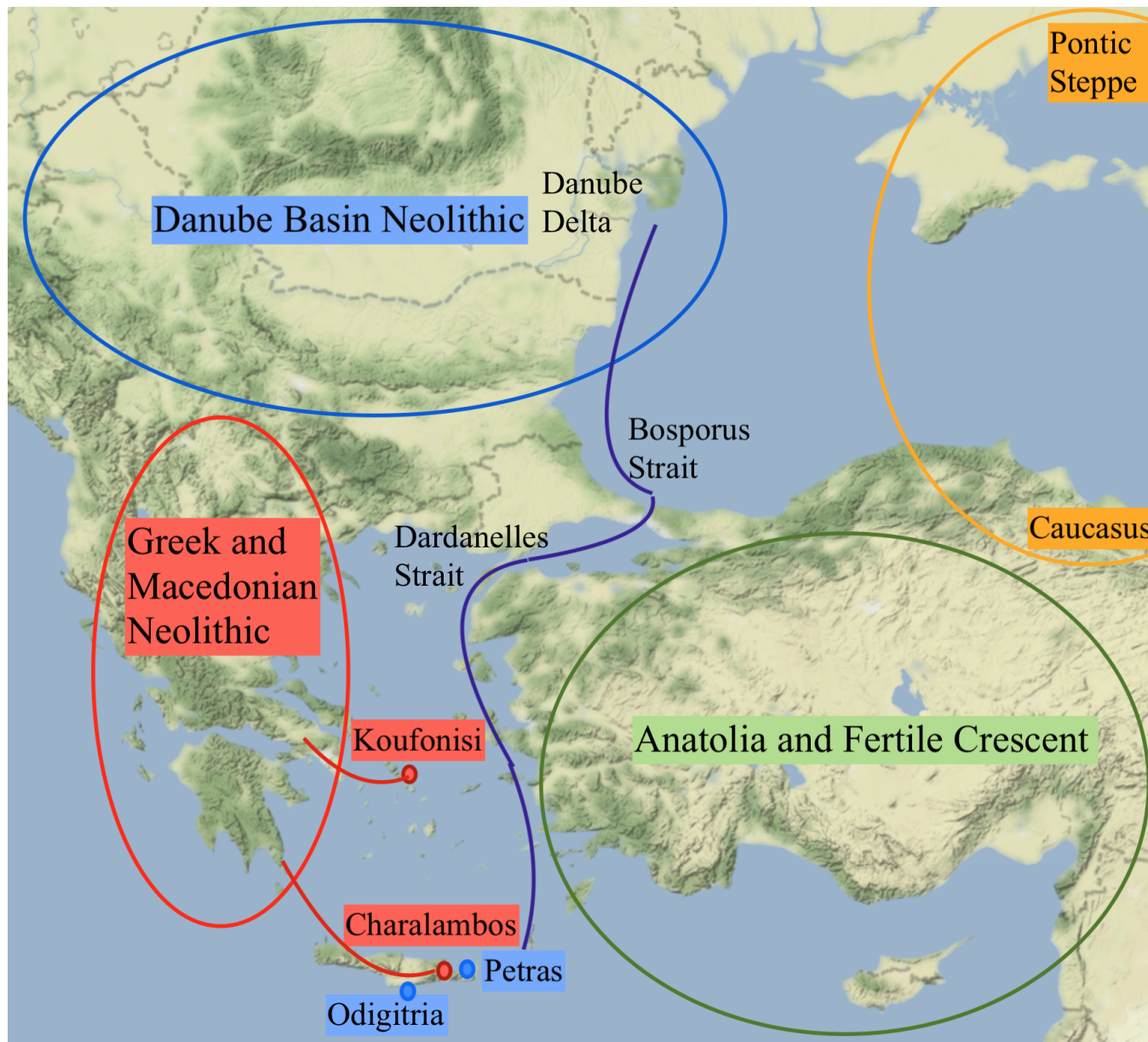


Figure 8: Some of the Aegean migration routes implied by the archaeological data mining.

as shown by both the cluster analysis in Section 4.2 and the PCA analysis in Section 4.3.

In addition, Haiga Triada is the largest source of Minoan Linear A documents. According to Revesz [20], the underlying language of Linear A belongs to the Finno-Ugric language family. Ancient topological names suggest that some people in the Black Sea area spoke Finno-Ugric languages before the arrival of Indo-Europeans [21]. At the Phaistos palace, which is also near Moni Odigitria, the Phaistos Disk was found, which shows another form of Minoan writing that is related to Cretan Hieroglyphs. These Minoan scripts share

the same underlying language with Linear A [17, 18]. The AIDA system provides an online, searchable library of Minoan inscriptions with their translations [24]. Cretan Hieroglyphs, Linear A and the Phaistos Disk script are members of the *Cretan Script Family*, which also includes the Linear B script that was used by the Mycenaeans [19]. The adoption of Linear A to Linear B was accompanied by a language change, where the signs were reinterpreted in the Mycenaean Greek language, which resulted in a change of the phonetic values of many of the signs. Daggumati and Revesz [6] use convolutional neural networks to extend the script comparison

to other ancient scripts, including the Indus Valley Script, which also has boustrophedon writing and a large number of allographs [7].

Possible Cause: This migration may have been prompted by a major climate change event, which is called the 4.2 kiloyear BP aridification event [8], when many areas of the world, including the Danube Basin, became dryer, making those areas infeasible for agriculture. Hence, farmers likely moved from those agricultural areas to the Aegean islands, which provided good fishing opportunities.

- (4) In the Late Bronze Age, around 1450 BC, the Mycenaeans conquered Crete. They established Knossos as the main center on Crete and modified the Minoan Linear A writing into Linear B, which records the earliest form of Greek language.

Possible Cause: The Santorini volcanic eruption c. 1600 BC resulted in a large tsunami that may have destroyed many of the Minoan coastal towns and the shipping fleet [1]. The volcanic eruption likely did not affect the Mycenaeans as negatively as the Minoans because while the tsunami hit northern Crete, it avoided the Argolis area, where the Mycenaeans lived. Hence the Mycenaean civilization could recover faster than the Minoan civilization after this catastrophe. As a result, the Mycenaeans became relatively stronger. Their surviving ships may have taken over some of the commercial activities of the Minoans when the latter's fleet was destroyed.

6 CONCLUSION AND FUTURE WORK

Most of the farmers in the Danube Basin had ancestry from Neolithic Anatolia. We are the first to test autosomal DNA for source ancestry from both populations simultaneously. The finding that the Neolithic Danube Basin is closer to the Moni Odigitria and Petras samples than to Neolithic Anatolian samples is surprising and contradicts the earlier hypothesis of Lazaridis et al. [15], which we quoted in the introduction. It also clarifies the confusion that Clemente et al. [5] introduced by grouping these two groups together into a "Neolithic European" category.

Archaeogenetic data mining cannot give a full answer to the question of what language was spoken by various groups in South-eastern Europe. Our archaeogenetic study suggests that the Early Minoan language may be related to some language spoken in the Greek Peloponnese peninsula around 5000 years ago, and the Middle Minoan language may be related to some language spoken in the Danube Basin area about 4200 years ago. The linguistic identification of the underlying language of Minoan Linear A as a Finno-Ugric language [20] tends to support our autosomal data mining results because part of the Danube Basin and the western Black Sea areas likely were Finno-Ugric language areas before the arrival of the Indo-Europeans. Some Neolithic Danube Basin archaeological sites contain undeciphered inscriptions in a script that resembles the Linear A script. That raises the possibility that the Linear A script was brought to Crete from the Danube Basin. However, more research is needed regarding the relationship between the Danube Basin script and the Minoan scripts.

REFERENCES

- [1] J. Antonopoulos. 1992. The great Minoan eruption of Thera volcano and the ensuing tsunami in the Greek Archipelago. *Natural Hazards*, 5, 153?168.
- [2] M. Bernal. 2006. *Black Athena: The Afroasiatic Roots of Classical Civilization: The Linguistic Evidence*, vol. III, Rutgers University Press, New Brunswick, NJ, USA.
- [3] G. Campbell-Dunn. 2014. *Who were the Minoans?: An African Answer*. BookWhirl Publishing: Green Bay, WI, USA.
- [4] V. C. Childe. 1922. The East-European relations of the Dimini Culture, *The Journal of Hellenic Studies*, vol. 42, no. 2, 254-275.
- [5] F. Clemente, M. Unterländer, O. Dolgova, C. Eduardo, C., G. Amorim, F. Coroado-Santos, S. Neuenschwander, E. Ganiatsou, M. I. Cruz Dávalos, L. Anchieri, F. Michaud, L. Winkelbach, J. Blöcher, Arizmendi Cárdenas, B. Sousa da Mota, E. Kalliga, A. Soules, I. Kontopoulos, G. Karamitrou-Mentessidi, O. Philaniotou, A. Sampson, D. Theodorou, M. Tsipopoulou, I. Akamatis, P. Halstead, K. Kotsakis, D. Urem-Kotsou, D. Panagiotopoulos, C. Ziota, S. Triantaphyllou, O. Delaneau, J. D. Jensen, J. V. Moreno-Mayar, J. Burger, V. C. Sousa, O. Lao, A.-S. Malaspinas, and C. Papageorgiou. 2021. The genomic history of the Aegean palatial civilizations, *Cell*, 184, 1-22.
- [6] S. Dagumati and P. Z. Revesz. 2018. Data mining ancient script image data using convolutional neural networks. In *Proceedings of the 22nd International Database Engineering and Applications Symposium*, pages 267-272. ACM Press.
- [7] S. Dagumati and P. Z. Revesz. 2021. A method of identifying allographs in undeciphered scripts and its application to the Indus Valley Script. *Humanities and Social Sciences Communications*, 8, 50.
- [8] P. B. deMenocal. 2001. Cultural responses to climate change during the late Holocene. *Science*, 292, 5517, 667-673.
- [9] E. Ehler, J. Novotný, A. Juras, M. Chyleński, O. Moravčík, and J. Pačes. 2019. A database of ancient human mitochondrial genomes, *Nucleic Acids Research*, 47, D1, p. D29-D32.
- [10] A. Evans. 1935. *The Palace of Minos at Knossos*, vols. I-IV, MacMillan and Co, London, UK.
- [11] M. Gimbutas. 1989. *The Language of the Goddess*, Thames & Hudson Inc., New York, NY, USA.
- [12] C. H. Gordon. 1966. *Evidence for the Minoan Language*, Ventnor Publ., Ventnor, NJ, USA.
- [13] H. Haarmann. 2014. *Roots of Ancient Greek Civilization: The Influence of Old Europe*, McFarland & Co. Publishing, Jefferson, NC, USA.
- [14] J. R. Hughey, P. Paschou, P. Drineas, D. Mastropaolo, D. M. Lotakis, P. A. Navas, M. Michalodimitrakis, J. A. Stamatoyannopoulos, and G. Stamatoyannopoulos. 2013. A European population in Minoan Bronze Age Crete, *Nature Communications*, 4, p. 1861.
- [15] J. Lazaridis, A. Mittnik, N. Patterson, S. Mallick, N. Rohland, S. Pfrengle, A. Furtwängler, A. Peltzer, C. Posth, A. Vasilakis, P. J. P. McGeorge, E. Konsolaki-Yannopoulou, G. Korres, H. Martlew, M. Michalodimitrakis, M. Özait, N. Özait, A. Papathanasiou, M. Richards, S. A. Roodenberg, Y. Tzedakis, R. Arnott, D. M. Fernandes, J. R. Hughey, D. M. Lotakis, P. A. Navas, Y. Maniatis, J. A. Stamatoyannopoulos, K. Stewardson, P. Stockhammer, R. Pinhasi, D. Reich, J. Krause, and G. Stamatoyannopoulos. 2017. Genetic origins of the Minoans and Mycenaeans, *Nature*, 548, 214-218.
- [16] N. Marinatos. 2010. *Minoan Kingship and the Solar Goddess: A Near Eastern Koine*, University of Illinois Press, Champaign, IL USA.
- [17] P. Z. Revesz. 2016. A computer-aided translation of the Cretan Hieroglyph script. *International Journal of Signal Processing*, 1, 127-133.
- [18] P. Z. Revesz. 2016. A computer-aided translation of the Phaistos Disk. *International Journal of Computers*, 10, 94-100.
- [19] P. Z. Revesz. 2016. Bioinformatics evolutionary tree algorithms reveal the history of the Cretan Script Family. *International Journal of Applied Mathematics and Informatics*, 10, 67-76.
- [20] P. Z. Revesz. 2017. Establishing the West-Ugric language family with Minoan, Hattic and Hungarian by a decipherment of Linear A, *WSEAS Transactions on Information Science and Applications*, 14, 1, 306-335.
- [21] P. Z. Revesz. 2018. Spatio-temporal data mining of major European river and mountain names reveals their Near Eastern and African origins, *22nd European Conference on Advances in Databases and Information Systems*, Springer LNCS 11019, p. 20-32.
- [22] P. Z. Revesz. 2019. Art motif similarity measure analysis: Fertile Crescent, Old European, Scythian and Hungarian elements in Minoan culture, *WSEAS Transactions on Mathematics*, 18, 264-287.
- [23] P. Z. Revesz. 2019. Minoan archaeogenetic data mining reveals Danube Basin and western Black Sea littoral origin, *International Journal of Biology and Biomedical Engineering*, 13, 108-120.
- [24] P. Z. Revesz, M. P. Rashid, Y. Tuyishime. 2019. The design and implementation of AIDA: Ancient Inscription Database and Analytics system, *Proc. 23rd Int. Database Engineering and Applications Symposium*, ACM Press, pp. 292-297.
- [25] Vahaduo tools for G25 admixture analysis, available at: <http://g25vahaduo.genetics.ovh>.

MANTIS: Multiple Type and Attribute Index Selection using Deep Reinforcement Learning

Vishal Sharma
Utah State University
Department of Computer Science
Logan, Utah, USA
vishal.sharma@usu.edu

Curtis Dyreson
Utah State University
Department of Computer Science
Logan, Utah, USA
curtis.dyreson@usu.edu

Nicholas Flann
Utah State University
Department of Computer Science
Logan, Utah, USA
nick.flann@usu.edu

ABSTRACT

DBMS performance is dependent on many *parameters*, such as index selection, cache size, physical layout, and data partitioning. Some combinations of these parameters can lead to optimal performance for a given workload but selecting an optimal or near-optimal combination is challenging, especially for large databases with complex workloads. Among the hundreds of parameters, index selection is arguably the most critical parameter for performance. We propose a self-administered framework, called the *Multiple Type and Attribute Index Selector* (MANTIS), that automatically selects near-optimal indexes. The framework advances the state-of-the-art index selection by considering both multi-attribute and multiple types of indexes within a bounded storage size constraint, a combination not previously addressed. MANTIS combines supervised and reinforcement learning, a Deep Neural Network recommends the type of index for a given workload while a Deep Q-Learning network recommends the multi-attribute aspect. MANTIS is sensitive to storage cost constraints and incorporates noisy rewards in its reward function for better performance. Our experimental evaluation shows that MANTIS outperforms the current state-of-art methods by an average of 9.53% *QphH@size*.

CCS CONCEPTS

• **Information systems** → **Autonomous database administration; Database management system engines; Database design and models;**

KEYWORDS

Database Index Selection, Markovian Decision Process, Deep Reinforcement Learning, Priority Experience Replay

ACM Reference format:

Vishal Sharma, Curtis Dyreson, and Nicholas Flann. 2021. MANTIS: Multiple Type and Attribute Index Selection using Deep Reinforcement Learning. In *Proceedings of 25th International Database Application & Engineering Symposium, Montreal, Canada, July, 2021 (IDEAS 2021)*, 9 pages. <https://doi.org/10.1145/3472163.3472176>

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

IDEAS 2021, July, 2021, Montreal, Canada

© 2021 Copyright held by the owner/author(s).

ACM ISBN 978-1-4503-8991-4/21/07.

<https://doi.org/10.1145/3472163.3472176>

1 INTRODUCTION

The performance of a database application is critical to ensuring that the application meets the needs of customers. An application's performance often depends on the speed at which *workloads*, i.e., a sequences of data retrieval and update operations are evaluated. A database can be *tuned* to improve performance, e.g., by creating an index or increasing the size of the buffer. Figure 1(b) shows the impact of just two configuration parameters, Memory Size and Buffer Size, on workload performance. We observe that choosing an optimal combination can enhance performance significantly. Currently, a database administrator (DBA) manually tunes configuration parameters by monitoring performance over time and adjusting parameters as needed. Nevertheless, the growing number of configuration parameters, as shown in Figure 1(a), has increased the complexity of manually tuning performance.

For many queries, e.g., range and lookup queries, a database index significantly reduces query time. An index can be created on a single column or several columns of a database table. There are also different *types* of indexes, for instance, Postgres has six index types: B-tree, Hash, GiST, SP-GiST, GIN and BRIN. One possible solution is to create all possible indexes for a database. However, this approach is infeasible due to a large number of potential indexes, e.g., for a single kind of index (B-tree) on a table with N columns, there are $2^N - 1$ potential (multi-column) indexes. There are two other reasons why it is crucial to limit the number of indexes in a database. First, there are space considerations. An index occupies space in secondary storage that increases the (stored) size of a database. Second, indexes slow down data modification since modifications need to update both the data and the index. Creating too many indexes can decrease the throughput and latency of a database. Hence the set of indexes created for a database should be *parsimonious*, while too few indexes may slow query evaluation, too many may increase space cost and slow down data modification.

The *index recommendation problem* can be defined as finding a set of indices that minimizes the time taken to evaluate a workload and the amount of storage used. Finding a set of indexes that minimizes the cost is a *combinatorial optimization problem*, and adding a disk size constraint makes this problem a *constrained combinatorial optimization problem*. Finding an optimal solution for such a problem is *NP-hard* [22]. Recently with the advancement of Reinforcement Learning (RL) [13], it is being utilized to find approximate solutions for large scale combinatorial optimization problems such as for vehicle routing [19], directed acyclic graph discovery [37], and the traveling salesman problem [16]. RL has also shown that it can learn complex database tasks with an ample search space, such as

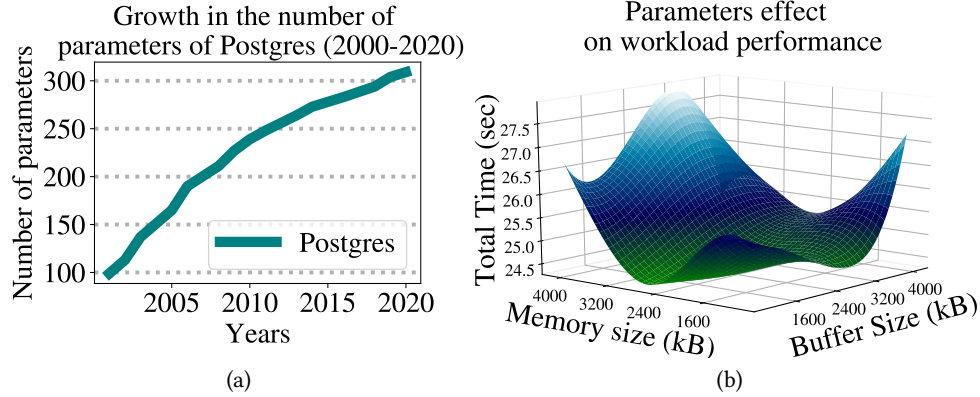


Figure 1: Figure (a) shows number of parameters in Postgres (2000-2020). Figure (b) shows effect on a workload performance using different parameter settings

learning optimal parameter configuration for a workload [26], and join order selection [17, 33].

In this paper, we apply RL to maximize query performance by learning which indexes would be the best to create for a given workload. There has been previous research on index recommendation using Reinforcement Learning [1, 2, 9, 14, 24, 31]. However, previous research has been limited in one of the three ways. First, previous research has focused on one type of index, B-Tree, but DBMSs typically support several types, e.g., Postgres has six types. Second, most previous research has investigated creating only single attribute indexes, but multi-attribute indexes can improve the performance of many queries. Third, previous research has not considered a constraint on storage size, that is, the approaches are *non-parsimonious* and allow the creation of more indexes than needed (there is no penalty for creating too many indexes). A generic framework that captures multi-attribute and different types of indexes is an open research problem. We propose an end-to-end index recommendation system that we call the *Multiple Type and Attribute Index Selector* (MANTIS). MANTIS can learn to recommend multi-attribute indexes of different types within a given size constraint.

This paper makes the following contributions.

- We formulate the *Index Recommendation Problem* as a *Markovian Decision Process*. We design our reward function using disk size constraint to limit the total index size.
- We propose an end-to-end multi-attribute and multi-index type recommendation framework. Our framework, MANTIS, uses Deep Neural Networks and Deep Q-Learning Networks for recommendations.
- We perform extensive experiments on MANTIS and compare results with current state-of-the-art methodologies on two different datasets.

This paper is organized as follows, the next section presents related work. Section 3 gives a precise formulation of the index recommendation problem while Section 4 describes solution to the problem using MANTIS. We present the evaluation of the performance of MANTIS in Sections 5 and 6. Section 7 presents conclusions and future work.

2 RELATED WORK

In this section, we discuss related work in the field of index recommendation using reinforcement learning. We identify the limitations of previous work and outline open area of research that this paper addresses.

A database can be tuned using either *external* or *internal* methods [7, 11, 21, 30]. An external tuning method uses an API to configure a DBMS, while an internal method embeds tuning algorithms in the DBMS. External tuning is widely preferred because it is generally applicable. Internal tuning needs access to DBMS internals, which may be proprietary or dependent on a particular DBMS software architecture. Most of the approaches discussed here are external. Internal tuning is primarily industry-based, e.g., the Oracle9i optimizer.

In recent years, Reinforcement Learning (RL) has become a popular external tuning method and has been used to optimize join order [17, 29, 33], in query optimization [8, 18, 20], for query scheduling [34], to self-tune databases [12, 35, 36] and to improve data partitioning [3, 5, 32]. For the *index selection problem*, Basu *et al.* [1] proposed a tuning strategy using Reinforcement Learning. They formulate the index selection problem as a Markovian Decision Process and use a state-space reduction technique to scale their algorithm for larger databases and workloads. Sharma *et al.* [26] proposed *NoDBA* for index selection. NoDBA stacks the workload and potential indexes as input to the neural network and uses Deep Reinforcement Learning with a custom reward function for index recommendation. Both of the above approaches consider only single attribute indexes and a single kind of index. They are unable to recommend multi-attribute indexes. Welborn *et al.* [31] introduced latent space representation for workload and action spaces. This representation enables them to perform other tasks, e.g., workload summarization and analyzing query similarity. They used a variant of DQN with dueling called BDQN (Branched Deep Q-Network) for learning index recommendation. Licks *et al.* [14, 15] introduced *SmartIX* where they use Q-Learning for index recommendation. In their approach, they learn to build indexes over multiple tables in a database. They also evaluate *SmartIX* using the standard metrics *QphH@size*, *power@size* and *throughput@size*. They use *QphH@size* in their reward function, which makes the evaluation

process slow, and in some cases, it may take several days of computation, which impacts the scalability of their approach. Kllapi *et al.* [6] propose a linear programming-based index recommendation algorithm. Their approach identifies and builds indexes during idle CPU time to maximize CPU utilization without affecting performance. Lan *et al.* [9] propose an index advisory approach using heuristic rules and Deep Reinforcement Learning. Sadri *et al.* [24] utilizes Deep Reinforcement Learning to select indexes for cluster databases. The previous work [9] and [24] evaluated the performance of a selected index by observing reduction in query execution cost. We believe that evaluating indexes based on execution cost is not ideal because it only measures the increase in read speed. The recommended indexes for modern databases must also be evaluated for durability (*power@size*) and processing capability (*throughput@size*). The above approaches have primarily focused on recommending B-tree indexes. By focusing on a single type of index, such approaches lack the capability of utilizing other types of indexes like BRIN or Hash to improve query performance. There are no previous approaches for performing an end-to-end index recommendation for both multi-attribute and multi-type index selection, which is the focus of this paper. Moreover, most previous approaches do not support a storage space constraint.

3 PROBLEM FORMULATION

The index recommendation problem is to select a set of indexes that minimizes the time to evaluate a workload and the amount of storage needed.

A workload W is a set of SQL queries Q_1, Q_2, \dots, Q_m . An index configuration I is a set of indexes. We calculate the cost of workload evaluation on database D using the cost of evaluating each query given by $Cost(Q_j, I, D)$. The cost of a workload can be described as follows.

$$Cost(W, I, D) = \sum_{j=1}^m Cost(Q_j, I, D)$$

Note that the workload cost does not weigh queries differently, though we could trivially include such weights by replicating individual queries in a workload. The index selection problem is to find a set of indexes $I_{optimal}$ that minimizes the total cost of workload $Cost(W, I, D)$ and has a storage cost of at most C .

$$I_{optimal} = \min_{S(I^*) \leq C} Cost(W, I^*, D)$$

In this equation, $S(I^*)$ is the total storage space cost of the set of indexes I^* .

4 MANTIS FRAMEWORK

We designed and built our framework using Deep Neural Networks (DNN) and Deep Q-Learning Networks (DQN). In order to select a suitable set of index types for a workload, our first research goal is *index type selection*. The second research goal is the *index recommendation* to pick possible (single/multiple) attributes for the index. Our framework, research goals, and the challenges we overcome are described in this section.

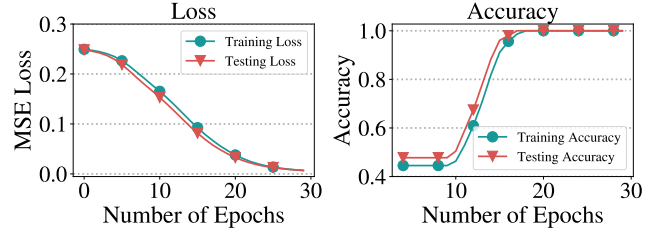


Figure 2: DNN training and testing performance

4.1 Index Type Selection

Almost all DBMSs have several types of indexes. Though the types vary across DBMSs, common index types include *B-tree index* (which includes all B-tree variants and is often the default index type), *block range index* (BRIN), which is helpful in range queries, and *hash index*, which improves the performance of hash joins and point queries. Spatial indexes, such as R-tree indexes, are also commonly available. We trained a Deep Neural Network (DNN) to choose the best types of indexes for a given workload. The DNN models take workload as an input and predict potential index types.

DNN model: We convert SQL queries in a workload to a vector representation using feature extraction. Specifically, we extract features describing different query types for B-Tree, BRIN, Hash, and Spatial.

- **feature_1:** Describes the count of each operator used in a query. The operators we search for are $[>, <, =, \leq, \geq]$. This feature helps us identify queries searching for equality, range, or general. In case of equality, a Hash index would be preferable, and for queries based on $[\leq, \geq]$ a BRIN index would be preferred over B-Tree.
- **feature_2:** The number of columns mentioned in a query.
- **feature_3:** The number of conjunctions/disjunctions $['and', 'or']$ in a query.
- **feature_4:** To identify spatial queries we extract certain keywords from the query $['.geom', '.location', '.distance']$.

Using the above features, we pre-train our DNN model. We use a fully connected multi-layer network with three layers. The first and second layers consist of 32 neurons with *relu* activation, and the output layer consists of *num_classes* (in our case 4) with *sigmoid* activation. The mean squared error (*mse*) is used as the cost function; the number of epochs is 30 and 30% data for validation. The model is trained using *adam* optimizer with a learning rate of 0.001, and we use a learning rate decay (initial rate/epochs) for the stability of the network. We use data generated by our TPC-H random query generator to pre-train. The training dataset comprises 500 queries each for four different types of indexes in total to 2000 queries. The process of generating queries is explained in a later Section 5. We observe the loss and accuracy of our DNN model to be stable, and Figure 2 displays DNN model performance.

4.2 Index Recommendation

Index recommendation in MANTIS uses Deep Neural Networks for function approximation with the Q-Learning algorithm, also known as the Deep Q-Learning Network (DQN). The indexes are

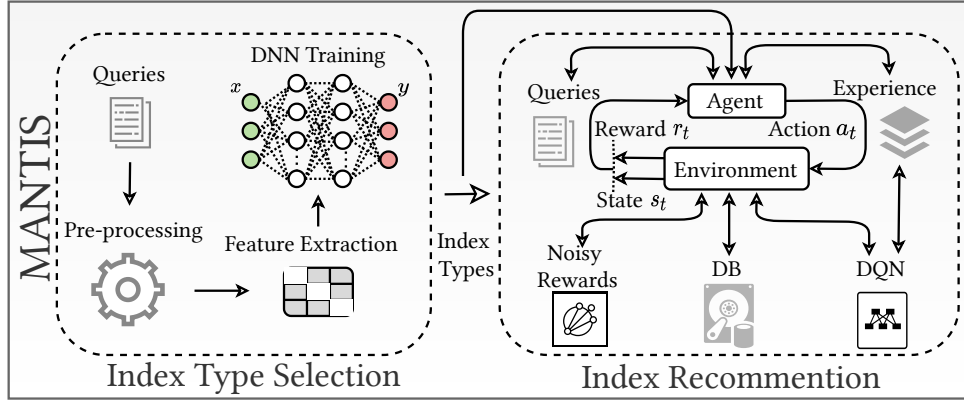


Figure 3: MANTIS framework using index type selection, and index recommendation.

interdependent, and this dependency can benefit performance. Such interdependency can be utilized for selecting a set of optimal indexes. Using the notion of interdependence, we formulate the index recommendation problem as a Markovian decision process (MDP). We extract relevant information from a database to define a *state*, *action*, and a $state_t$ to $action_t$ mapping at time t . We represent a *state* using existing indexes in a system and an *action* using a set of all possible indexes. Both the state and action are deterministic. In a conventional MDP, every action is mapped to a state and an associated reward. The goal of MDP is to reach the final state, maximizing cumulative rewards and identifying a *policy*, which is a state-action mapping that selects the appropriate action at a given state. The two fundamental methods to solve MDPs are *value iteration* and *policy iteration*. Value iteration uses the *value* of a state that quantifies amount of future rewards it can generate using the current *policy*, also known as *expected return*. Policy iteration uses the *policy* of a state-action pair that signifies the amount of current reward it can generate with an action at a state using a specific policy.

MANTIS uses a variant of value iteration called Q-Learning. Q-Learning is a value-based, temporal-difference reinforcement learning algorithm. In a Q-Learning algorithm, the agent learns from the history of environment interactions. Q-Learning uses an average of old and newer observations to update neural networks. It reduces temporal dependence due to random data sampling for training, leading to faster convergence. Our framework can also learn in a constrained environment, in our case, a storage size bound. The constraint is generic and could be used in conjunction with other variables, e.g., buffer size, memory size.

State and Action: An agent interacts with a *state* for the learning process. A *state* is the representation of the environment. In a state representation, the information given is available to the learning agent. It should be capable of accurately explaining the environment. We use a potential set of indexes as the state representation. The *action* space represents all possible actions that can be performed. We calculate the number of possible actions $N_{actions}$ using the following equation:

$$N_{actions} = (2^{N_{columns}} - 1) \times \text{len}(\text{index_type})$$

where, *index_type* is our DNN model predicting the possible index types to be used and $(2^{N_{columns}} - 1)$ is all possible combination of indexes in a table.

Reward Function: A reward function is another vital component of a reinforcement learning agent. We design our reward function based on workload cost estimation. We compare the cost of the index set against the set of all possible indexes as defined below:

$$rewards_{size} = \begin{cases} 1, & \text{index_size} < \text{max_allowed_size} \\ -1, & \text{otherwise} \end{cases}$$

$$r_t = \max \left(\frac{\text{no_index_cost}}{\text{selected_index_cost}} - 1, 0 \right) + rewards_{size} \quad (1)$$

where, the denominator is the workload cost with a selected set of indexes, and the numerator is workload cost with no indexes. We also use a reward for the storage size constraint.

Noisy Rewards: Inconsistent rewards can cause an agent's performance to degrade, and a learning agent will not maximize rewards. Though we use query cost estimate as a reward, previous research has shown the inefficacy of DBMS cost estimator [10]. To minimize the noise in the reward function, we perform a 1D convolution filter with a kernel size of five and use the filter in our reward function. Given an input vector, f of size k and convolution kernel g of size m , a 1D convolution filter can be formulated as follows:

$$(f * g)(j) = \sum_{i=1}^m (g(i) \cdot f(j - i + m/2)) / m$$

We experimented with several well-known noise-reducing filters, namely Exponential filter, Savitzky-Golay filter, and Kalman Filter. We found that these filters either over or underestimated rewards. The 1D convolution filter also tends to underestimate the cost. However, due to its fast computation time and streaming nature, it proved to be a feasible solution. We also added a spike filter to suppress extreme values.

DQN with Priority Experience Replay: The Priority Experience Replay (PER) has proven to be a very effective improvement over traditional sampling strategy in a DQN algorithm [25]. Rather

Algorithm 1 DQN with Priority Experience Replay ▷ (with hyperparameters initial values)

```

1: Initialize batch size  $N_s$  ▷ 32
2: Initialize number of Iterations  $I_s$  ▷ 100
3: Initialize length of a episode  $L_s$  ▷ 3-6
4: Initialize update target network frequency  $F_s$ 
5: Initialize priority scale  $\eta$  ▷ 1.0
6: Initialize priority constant  $\epsilon$  ▷ 0.1
7: Initialize network parameter  $\theta$ 
8: Initialize target network parameter  $\theta'$ 
9: Initialize learning rate  $\alpha$  ▷ 0.001
10: Initialize discount factor  $\beta$  ▷ 0.97
11: for  $l \in L_s$  do ▷ number of episodes
12:   Collect experiences  $(s, a, r, s', p)$  ▷ until minimum  $N_s$  size
13:   for  $i \in I_s$  do
14:     Sample  $N_s$  prioritized samples from buffer
15:     for  $n \in N_s$  do
16:        $y_i = r_i + \beta \max_{a'_i \in A} Q_{\theta'}(s'_i, a'_i)$ 
17:       if  $done == True$  then ▷ check for terminal state
18:          $\delta_i = |y_i - Q_{\theta}(s_i, a_i)|$  ▷ calculate TD Error
19:       end if
20:       end for
21:        $L(\theta) = \frac{1}{N_s} \sum_i (y_i - Q_{\theta}(s_i, a_i))^2$  ▷ calculate loss MSE
22:        $\theta = \theta - \alpha \nabla_{\theta} L(\theta)$  ▷ update network parameters
23:        $p_i = \frac{(|\delta_i| + \epsilon)^{\eta}}{\sum_j (|\delta_j| + \epsilon)^{\eta}}$  ▷ calculate and update samples priority
24:     end for
25:     if  $l \bmod F_s$  then
26:        $\theta' = \theta$  ▷ update target network
27:     end if
28:   end for

```

than uniform sampling, PER weighs the sample such that the sample producing high error will be drawn more frequently in training. The PER helps in reducing the overall bias and improved the performance of the network. We compute PER based on Temporal Difference (TD) error. The TD error (δ) is computed using equation below:

$$\delta_i = |y_i - Q_{\theta}(s_i, a_i)|$$

where, y_i is the target and $Q_{\theta}(s_i, a_i)$ is the estimate. The target (y_i) is computed using Bellman's optimality equation [27], as shown below.

$$y_i = r_i + \beta \max_{a'_i \in A} Q_{\theta'}(s'_i, a'_i)$$

where, r_i is the computed reward, $\max_{a'_i \in A} Q_{\theta'}(s'_i, a'_i)$ is the future rewards, and β is the discount factor. The priority (p_i) of the samples are computed using TD error by:

$$p_i = \frac{(|\delta_i| + \epsilon)^{\eta}}{\sum_j (|\delta_j| + \epsilon)^{\eta}}$$

To learn the weights of Neural Network parameters in DQN, we use stochastic gradient descent on MSE Loss $L(\theta)$ function.

DQN Agent training: The training procedure consists of $N_{episodes}$ episodes, and each episode has N_{index} steps where N_{index} represents the maximum number of indexes. During an episode, the agent performs an action based on a policy and collects rewards for the selected action, known as an *experience*. These experiences are stored in a buffer, called the *Replay Buffer*, for sampling (*priority based*) and training. The cost of an action is calculated by its effect

on the total workload cost of retrieval using the rewards function from Equation 1. The computed rewards are adjusted using the 1D convolution filter. At the end of this step *state*, *action*, *rewards* are returned. During training we use the a configuration batch size of 32, consisting of 100 episodes, a maximum index number of N_{index} : 3–6, a priority scale of η : 0.97, a learning rate of α : 0.001, a discount factor of β : 0.97, and a storage bound of 10-20MB. Our complete algorithm with hyperparameter values is shown in Algorithm 1, and the framework is depicted in Figure 3.

5 EXPERIMENTAL SETUP

We perform experiments on two datasets, a standard database benchmark TCP-H [28] and a real time dataset IMDB [10]. We use PostgreSQL as a choice for our database. We create an OpenGym environment for command-based database interaction. All experiments were performed on a computer with Intel i7 5820k, Nvidia 1080ti, 32GB of RAM running Ubuntu 18.04 OS. We use Python 3.7 and libraries (powa, gym, psycopg2, sklearn, TensorFlow, Keras) to write and train our framework. The DNN and DQN were trained on Nvidia 1080ti (3584 CUDA cores and 11GB DDR5 RAM) with CUDA and cuDNN configured for performance enhancement.

(1) **TPC-H:** TPC is the most well-known and most widely-used family of database benchmarks. TPC-H is the benchmark for decision support systems. There exists a set of 22 TPC-H query templates. The set of query templates are majorly used for benchmarking database systems. However, they are not specific for index selection benchmarking. With this in mind, we generate a dataset of 120k tuples using TPC-H and randomly generate queries as follows: we randomly select columns and a value from a table. We then randomly select an operator [$>$, $<$, $=$] and predicate [and, or]. We create four different sets of queries by randomly selecting between one and four columns (1C, 2C, 3C, 4C). The variety of queries assists in the validation of our framework's efficacy. We generate 100 queries for each TPC-H experiment. We use only 120k rows in this experiment. In our future work, we plan to focus on larger-scale index selection. Our purpose of generating random queries from TPC-H is to simulate different types of environments and observe the performance of MANTIS compared to baseline. This experiment uses single column (1C) and multi-column indexes (2C, 3C, and 4C) simulation.

Few randomly generated queries used in our experiments:

```

1C: SELECT COUNT(*) FROM LINEITEM WHERE L_TAX < 0.02
2C: SELECT COUNT(*) FROM LINEITEM WHERE L_ORDERKEY <
    11517219 OR L_TAX < 0.02
3C: SELECT COUNT(*) FROM LINEITEM WHERE L_SUPPKEY < 18015
    AND L_PARTKEY > 114249 AND L_TAX > 0.06
4C: SELECT COUNT(*) FROM LINEITEM WHERE L_ORDERKEY = 8782339
    AND L_TAX = 0.01 AND L_PARTKEY = 264524 AND L_SUPPKEY >
    14028

```

(2) **IMDB:** IMDB is an extensive database of movie-related data. There are 21 tables, with a few large tables such as cast_info table, which has 36 million records, and the movie_info table, which has 15 million records. It has 113 computationally intensive SQL queries with multi-joins. We randomly and equally divide the queries into three stages with 37 (Stage 1), 38 (Stage 2), and 38 (Stage 3) queries.

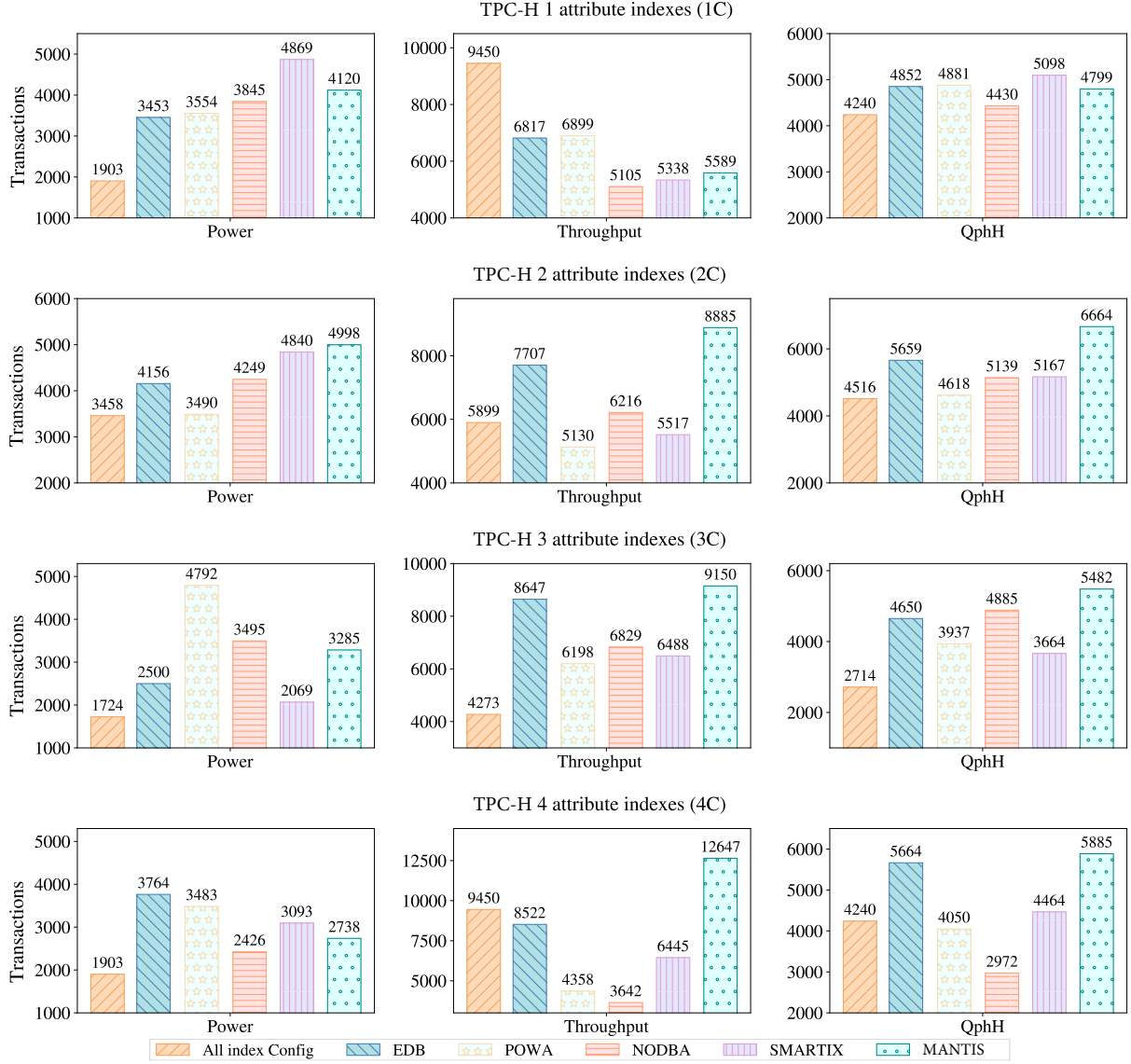


Figure 4: Power, Throughput and QphH of index recommendation methods and there comparison in different scenarios on TPC-H dataset.

The aim of separating queries is to create more real-time scenarios for framework validation. We evaluate our framework on all three stages by selecting indexes and comparing performance with baseline.

5.1 Performance Metric

We measure performance using standard DBMS metrics, such as *Power@size*, *Throughput@Size*, and *QphH@Size*.

Power@Size tests the durability of the indexes chosen for inclusion and deletion of documents throughout the database. It includes a variety of steps, including (1) a refresh function RF1 that inserts 0.1% of the table's data, (2) the execution of a single stream of

queries, and (3) the time taken by the RF2 refresh feature, which deletes 0.1% of the table's records at random. The following equation is used to calculate the metric:

$$Power@Size = \frac{3600}{N_q \sqrt{(\alpha_{i=1}^{N_q} ET(i)) \times (\alpha_{j=1}^2 RF(j))}} \times Scale\ Factor$$

where, N_q is the number of queries, $ET(i)$ is execution time for each query i , $RF(j)$ is the time taken by the two refresh functions, and $Scale\ Factor$ is the factor of database size used from TPC-H and IMDB.

Throughput@Size measures the processing capability of a system (disk I/O, CPU speed, memory bandwidth, BUS speed, etc.). It

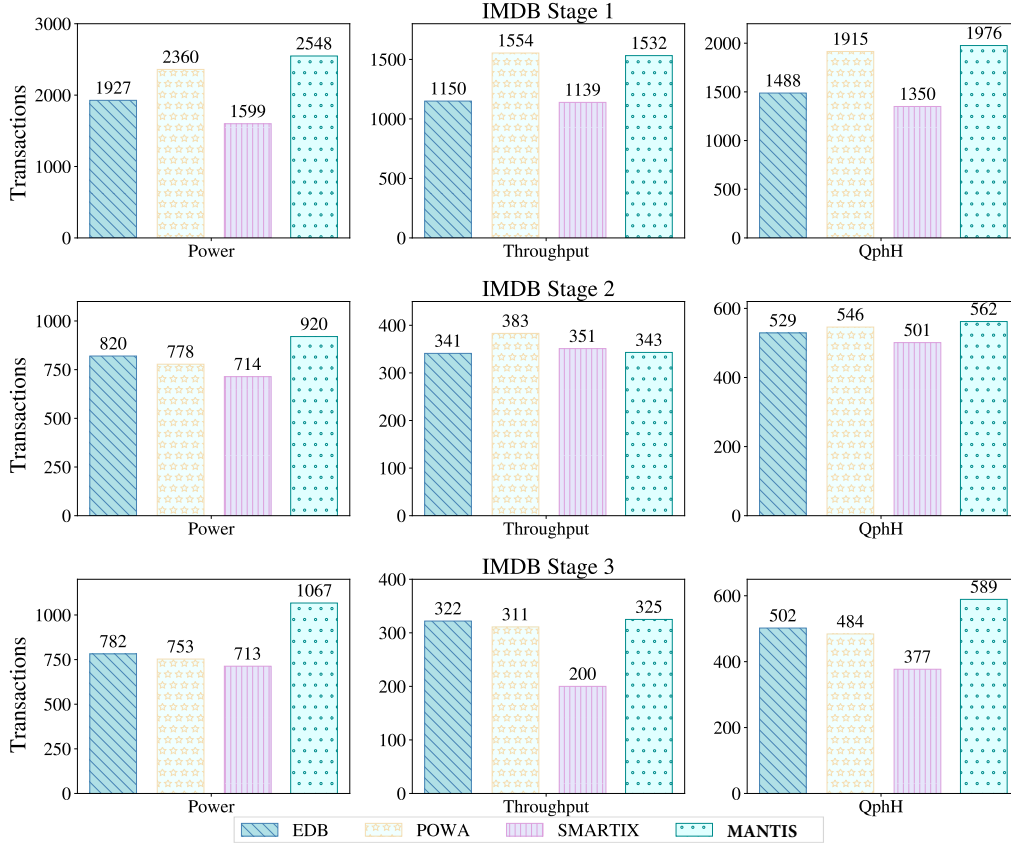


Figure 5: Power, Throughput and QphH of index recommendation methods and there comparison in different scenarios on IMDB dataset.

is computed using the equation below:

$$Throughput@Size = \frac{QueryStream \times N_q}{TotalTime} \times 3600 \times ScaleFactor$$

where, *QueryStream* is the number of query streams (for our experiments we used only a single stream) and *TotalTime* is the time taken to execute all queries for all streams.

QphH@Size measures multiple aspects of database performance. It measures the query processing power and throughput when queries are from multiple streams. The Query-per-Hour Performance Metric (QphH) is calculated using Power@Size and Throughput@Size, as shown below:

$$QphH@Size = \sqrt{Power@Size \times Throughput@Size}$$

Experimental Design: In the TPC-H 1C with only a single attribute index selection, there are 16 states (number of columns from LINEITEM). We randomly select four columns for multi-attribute index selection, and for 2, 3, and 4 column indexes, there are 15, 80, and 255 states, respectively. For the IMDB dataset, we use all tables and columns in index selection. We use only single column indexes, and the state space consists of 108 states. The number of states is crucial for the initialization of action and state space of the RL agent. The baseline is evaluated on the database with identical records and workload.

5.2 Baselines

There are several recent approaches for index selection using reinforcement learning [9, 14, 24]. These methods, in general, utilize Deep Q-Networks with Priority Experience Replay in their approach. However, only SmartIX [14] evaluated its performance with other existing and state-of-the-art methods. The other methods [9, 24] do not compare their performance with any other state-of-the-art methods. Moreover some [9, 24] approaches measure their performance only based on reduction in query execution time. The selected indexes can not be justified as optimal by such an evaluation. Since in modern database system indexes are updated frequently (updates and writes) and a better performance evaluation is performed by evaluating latency and throughput of indexes using a standard database metric *Power@Size*, *Throughput@Size*, and *QphH@Size*. With these reasons in consideration, we select SmartIX [14] as one of the baselines. Overall, We select two enterprise-based solutions and two index selection using RL, which we re-implemented based on information provided in their research papers.

POWA [23]: The PostgreSQL Workload Analyzer is an optimization tool for PostgreSQL. It collects various statistical data from a database and suggests indexes to optimize the workload.

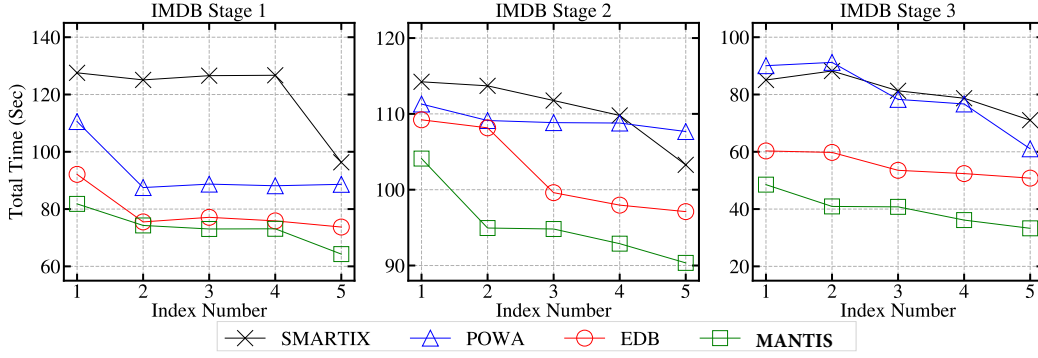


Figure 6: Workload execution time with respect to selected indexes on IMDB dataset using MANTIS and baselines.

EDB [4]: EnterpriseDB’s Postgres advanced server is designed to customize, tune, and handle massive Postgres database deployments. As a benchmark, we use their index advisor tool.

NODBA [26]: We re-implement NODBA based on details from paper. The authors use DQN for index recommendations. They use query and index configuration as input to the Neural Network.

SMARTIX [14]: We re-implement SMARTIX based on details provided in the paper. The authors use $QphH@Size$ as a reward function and Q-Learning for selecting indexes.

All Index: We create all possible indexes for the database and use that as a benchmark. For experiments with IMDB, we do not use **All Index** due to the large index space size and also **NODBA**, it does not support multi-table index selection.

6 RESULTS

In this section, we compare and discuss the performance of MANTIS with baselines on TPC-H and IMDB datasets. We evaluate our framework on different scenarios specifically, single attribute index, multi attribute index, and index selection on real-time. We also measure and compare the workload execution time.

6.1 TPC-H

(1) **First Scenario (1C - single attribute indexes)**: We calculate $Power@Size$, $Throughput@Size$ and $QphH@Size$ for all of the baseline systems. We observe comparable performance among most of the baseline and MANTIS. Specifically, SMARTIX performs the best in this setting, followed by POWA and EDB. Most of the baselines are designed and tuned specially for single index selection scenarios.

(2) **2C, 3C and 4C Scenario (multi-attribute indexes)**: We use both one and two columns indexes for 2C. We observe that MANTIS performs best with 17.7% $QphH$ improvement to the second baseline. We use 1, 2, and 3 column indexes for the 3C scenario. Our framework shows a 12.1% $QphH$ improvement to the second baseline. We use 1,2,3 and 4 columns for the 4C scenario. We observe a 4% $QphH$ improvement of our framework over the best baseline. All the results are shown in Figure 4.

We observe that our framework outperforms other baselines on the TPC-H dataset in most (3/4) of the scenarios. Next, we measure the performance on the IMDB dataset.

6.2 IMDB

We observe MANTIS outperformed other baseline systems in all three stages, as shown in Figure 5. Specifically, there is 3.19%, 2.9%, and 17.3% of $QphH$ improvement to the best baseline at Stage 1, Stage 2, and Stage 3, respectively. Our framework can learn complex index selection where other baselines struggle. Overall, we observe that our framework outperforms other baselines (3/4) on TPC-H and IMDB (3/3) datasets. The runtime of MANTIS for TPC-H is about 2 hrs and for IMDB is about 6 hrs.

To better understand the results from IMDB, we design an experiment to answer: *how effective are selected indexes?* Ideally, we would like to observe a drop in the workload’s overall execution time when indexes are created. We execute the benchmark and measure performance after every index creation. The results are shown in Figure 6. The index selected using MANTIS took the least time in all stages. We also observe that the first index selected by MANTIS is optimal in all stages. There is also a steady reduction in workload execution costs, which is ideal.

7 CONCLUSION AND FUTURE WORK

This paper presents MANTIS, a framework to recommend indexes for enhancing the efficiency of a query workload. We propose an end-to-end framework, MANTIS, for index recommendation in a database. Our implemented framework uses a Deep Neural Network for index type selection and a Deep Q-Learning Network algorithm for multi-attribute index recommendation. Compared to previous methods, MANTIS can learn and propose single-attribute, multi-attribute, and multi-type indexes. We evaluate MANTIS with four other state-of-the-art methods using two standard benchmark datasets. We use standard DBMS performance metrics $Power@Size$, $Throughput@Size$ and $QphH@Size$ for evaluation. The experiments show that MANTIS can significantly outperform (6/7 cases) the state-of-the-art index recommendation methods.

In the future, we aim to build a self-managing database utilizing Machine and Reinforcement Learning techniques. We intend to carry out our objective in stages. In the first stage, we plan to extend our work MANTIS by adding more configuration parameters and evaluating our framework on larger and real-time environments. This future work is the first stage of evaluating Machine and Reinforcement Learning techniques to autonomously configure

a database at a large scale. In the second stage, we plan to apply machine learning to optimally modify or extend a set of indexes in response to a changing workload, in essence performing incremental index selection. In the third stage, we plan to extend incremental index selection to online selection of other configuration parameters. Our objective in each stage is to move one step closer to a self-managed database. Another direction of our future work is to study temporal index selection for temporal databases, which is an open problem.

ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation under Award No. DBI-1759965, *Collaborative Research: ABI Development: Symbiota2: Enabling greater collaboration and flexibility for mobilizing biodiversity data*. We would like to thank all the reviewers for providing valuable feedback that significantly improved the quality of our paper.

REFERENCES

- [1] Debabrota Basu, Qian Lin, Weidong Chen, Hoang Tam Vo, Zihong Yuan, Pierre Senellart, and Stéphane Bressan. 2015. Cost-Model Oblivious Database Tuning with Reinforcement Learning. In *DEXA (DEXA 2015)*. Springer-Verlag, Berlin, Heidelberg, 253–268.
- [2] Bailu Ding, Sudipto Das, Ryan Marcus, Wentao Wu, Surajit Chaudhuri, and Vivek R. Narasayya. 2019. AI Meets AI: Leveraging Query Executions to Improve Index Recommendations. In *SIGMOD*. 1241–1258.
- [3] Gabriel Campero Durand, Marcus Pinnecke, Rufat Piriyev, Mahmoud Mohsen, David Broneske, Gunter Saake, Maya S. Sekeran, Fabián Rodríguez, and Laxmi Balam. 2018. GridFormation: Towards Self-Driven Online Data Partitioning using Reinforcement Learning. In *aiDM'18*.
- [4] EDB. 2020. Enterprise Database (EDB): Power to Postgres. (2020). <https://www.enterprisedb.com/>
- [5] Benjamin Hilprecht, Carsten Binnig, and Uwe Röhm. 2019. Towards Learning a Partitioning Advisor with Deep Reinforcement Learning. In *Proceedings of the Second International Workshop on Exploiting Artificial Intelligence Techniques for Data Management (aiDM '19)*. Association for Computing Machinery, New York, NY, USA, Article 6, 4 pages. <https://doi.org/10.1145/3329859.3329876>
- [6] Herald Killapi, Ilia Pietri, Verena Kantere, and Yannis E Ioannidis. 2020. Automated Management of Indexes for Dataflow Processing Engines in IaaS Clouds. In *EDBT*. 169–180.
- [7] Tim Kraska, Umar Farooq Minhas, Thomas Neumann, Olga Papaemmanouil, Jignesh M Patel, Chris Ré, and Michael Stonebraker. 2021. ML-In-Databases: Assessment and Prognosis. *Data Engineering* (2021), 3.
- [8] Sanjay Krishnan, Zongheng Yang, Ken Goldberg, Joseph M. Hellerstein, and Ion Stoica. 2018. Learning to Optimize Join Queries With Deep Reinforcement Learning. *CoRR abs/1808.03196* (2018). <http://arxiv.org/abs/1808.03196>
- [9] Hai Lan, Zhifeng Bao, and Yuwei Peng. 2020. An Index Advisor Using Deep Reinforcement Learning. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2105–2108.
- [10] Viktor Leis, Andrey Gubichev, Atanas Mirchev, Peter Boncz, Alfons Kemper, and Thomas Neumann. 2015. How Good Are Query Optimizers, Really? *Proc. VLDB Endow.* 9, 3 (Nov. 2015), 204–215. <https://doi.org/10.14778/2850583.2850594>
- [11] Guoliang Li, Xuanhe Zhou, and Lei Cao. 2021. AI Meets Database: AI4DB and DB4AI. In *Proceedings of the 2021 International Conference on Management of Data (SIGMOD/PODS '21)*. Association for Computing Machinery, New York, NY, USA, 2859–2866. <https://doi.org/10.1145/3448016.3457542>
- [12] Guoliang Li, Xuanhe Zhou, Shifu Li, and Bo Gao. 2019. QTune: A Query-Aware Database Tuning System with Deep Reinforcement Learning. *Proc. VLDB Endow.* 12, 12 (Aug. 2019), 2118–2130. <https://doi.org/10.14778/3352063.3352129>
- [13] Yuxi Li. 2017. Deep Reinforcement Learning: An Overview. *CoRR abs/1701.07274* (2017). <http://arxiv.org/abs/1701.07274>
- [14] Gabriel Paludo Licks, Julia Colleoni Couto, Priscilla de Fátima Míche, Renata De Paris, Duncan Dubugras Ruiz, and Felipe Meneguzzi. 2020. SMARTIX: A database indexing agent based on reinforcement learning. *Applied Intelligence* (2020), 1–14.
- [15] Gabriel Paludo Licks and Felipe Meneguzzi. 2020. Automated database indexing using model-free reinforcement learning. *arXiv preprint arXiv:2007.14244* (2020).
- [16] Qiang Ma, Suwen Ge, Danyang He, Darshan Thaker, and Iddo Drori. 2019. Combinatorial Optimization by Graph Pointer Networks and Hierarchical Reinforcement Learning. (2019). [arXiv:cs.LG/1911.04936](http://arxiv.org/abs/1911.04936)
- [17] Ryan Marcus and Olga Papaemmanouil. 2018. Deep Reinforcement Learning for Join Order Enumeration. In *Proceedings of the First International Workshop on Exploiting Artificial Intelligence Techniques for Data Management (aiDM'18)*. ACM, New York, NY, USA, Article 3, 4 pages. <https://doi.org/10.1145/3211954.3211957>
- [18] Ryan Marcus and Olga Papaemmanouil. 2018. Towards a Hands-Free Query Optimizer through Deep Learning. *CoRR abs/1809.10212* (2018). [arXiv:1809.10212](http://arxiv.org/abs/1809.10212)
- [19] Mohammadreza Nazari, Afshin Oroojlooy, Lawrence Snyder, and Martin Takáč. 2018. Reinforcement learning for solving the vehicle routing problem. In *Advances in Neural Information Processing Systems*. 9839–9849.
- [20] Jennifer Ortiz, Magdalena Balazinska, Johannes Gehrke, and S Sathiya Keerthi. 2018. Learning state representations for query optimization with deep reinforcement learning. In *Proceedings of the Second Workshop on Data Management for End-To-End Machine Learning*. 1–4.
- [21] Andrew Pavlo, Matthew Butrovich, Ananya Joshi, Lin Ma, Prashanth Menon, Dana Van Aken, L. James Lee, and Ruslan Salakhutdinov. 2019. External vs. Internal: An Essay on Machine Learning Agents for Autonomous Database Management Systems. *IEEE Data Eng. Bull.* 42 (2019), 32–46.
- [22] Gregory Piatesky-Shapiro. 1983. The Optimal Selection of Secondary Indices is NP-Complete. *SIGMOD Rec.* 13, 2 (Jan. 1983), 72–75.
- [23] PowA-Team. 2020. PostgreSQL Workload Analyzer (Powa). (2020). <https://github.com/powa-team/powa>
- [24] Zahra Sadri, Le Gruenwald, and Eleazar Lead. 2020. DRLindex: Deep Reinforcement Learning Index Advisor for a Cluster Database. In *Proceedings of the 24th Symposium on International Database Engineering & Applications (IDEAS '20)*. Association for Computing Machinery, New York, NY, USA, Article 11, 8 pages. <https://doi.org/10.1145/3410566.3410603>
- [25] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. 2016. Prioritized Experience Replay. In *ICLR (Poster)*.
- [26] Ankur Sharma, Felix Martin Schuhknecht, and Jens Dittrich. 2018. The Case for Automatic Database Administration using Deep Reinforcement Learning. *CoRR abs/1801.05643* (2018). [arXiv:1801.05643](http://arxiv.org/abs/1801.05643)
- [27] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.
- [28] TPC-H. 2020. TPC-H Decision Support Benchmark. (2020). <http://www.tpc.org/tpch/>
- [29] Immanuel Trummer, Samuel Moseley, Deepak Maram, Saehan Jo, and Joseph Antonakakis. 2018. SkinnerDB: Regret-bounded Query Evaluation via Reinforcement Learning. *Proc. VLDB Endow.* 11, 12 (Aug. 2018), 2074–2077. <https://doi.org/10.14778/3229863.3236263>
- [30] Dana Van Aken, Andrew Pavlo, Geoffrey J. Gordon, and Bohan Zhang. 2017. Automatic Database Management System Tuning Through Large-Scale Machine Learning. In *Proceedings of the 2017 ACM International Conference on Management of Data (SIGMOD '17)*. Association for Computing Machinery, New York, NY, USA, 1009–1024. <https://doi.org/10.1145/3035918.3064029>
- [31] Jeremy Welborn, Michael Schaarschmidt, and Eiko Yoneki. 2019. Learning Index Selection with Structured Action Spaces. *arXiv preprint arXiv:1909.07440* (2019).
- [32] Zongheng Yang, Badrish Chandramouli, Chi Wang, Johannes Gehrke, Yinan Li, Umar Farooq Minhas, Per-Åke Larson, Donald Kossmann, and Rajeev Acharya. 2020. Qd-Tree: Learning Data Layouts for Big Data Analytics. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data (SIGMOD '20)*. Association for Computing Machinery, New York, NY, USA, 193–208. <https://doi.org/10.1145/3318464.3389770>
- [33] X. Yu, G. Li, C. Chai, and N. Tang. 2020. Reinforcement Learning with Tree-LSTM for Join Order Selection. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*. 1297–1308.
- [34] Chi Zhang, Ryan Marcus, Anat Kleiman, and Olga Papaemmanouil. 2020. Buffer Pool Aware Query Scheduling via Deep Reinforcement Learning. *arXiv preprint arXiv:2007.10568* (2020).
- [35] Ji Zhang, Yu Liu, Ke Zhou, Guoliang Li, Zhili Xiao, Bin Cheng, Jiashu Xing, Yangtao Wang, Tianheng Cheng, Li Liu, Minwei Ran, and Zekang Li. 2019. An End-to-End Automatic Cloud Database Tuning System Using Deep Reinforcement Learning. In *Proceedings of the 2019 International Conference on Management of Data (SIGMOD '19)*. Association for Computing Machinery, New York, NY, USA, 415–432. <https://doi.org/10.1145/3299869.3300085>
- [36] Ji Zhang, Ke Zhou, Guoliang Li, Yu Liu, Ming Xie, Bin Cheng, and Jiashu Xing. 2021. CDBTune+: An efficient deep reinforcement learning-based automatic cloud database tuning system. *The VLDB Journal* (2021), 1–29.
- [37] Shengyu Zhu, Ignavier Ng, and Zhitang Chen. 2020. Causal Discovery with Reinforcement Learning. In *International Conference on Learning Representations (ICLR)*.

Explainable Data Analytics for Disease and Healthcare Informatics

Carson K. Leung*
Department of Computer Science
University of Manitoba
Winnipeg, MB, Canada
kleung@cs.umanitoba.ca

Daryl L.X. Fung
Department of Computer Science
University of Manitoba
Winnipeg, MB, Canada

Thanh Huy Daniel Mai
Department of Computer Science
University of Manitoba
Winnipeg, MB, Canada

Joglas Souza
Department of Computer Science
University of Manitoba
Winnipeg, MB, Canada

Nguyen Duy Thong Tran
Department of Computer Science
University of Manitoba
Winnipeg, MB, Canada

Qi Wen
Department of Computer Science
University of Manitoba
Winnipeg, MB, Canada

ABSTRACT

With advancements in technology, huge volumes of valuable data have been generated and collected at a rapid velocity from a wide variety of rich data sources. Examples of these valuable data include healthcare and disease data such as privacy-preserving statistics on patients who suffered from diseases like the coronavirus disease 2019 (COVID-19). Analyzing these data can be for social good. For instance, data analytics on the healthcare and disease data often leads to the discovery of useful information and knowledge about the disease. Explainable artificial intelligence (XAI) further enhances the interpretability of the discovered knowledge. Consequently, the explainable data analytics helps people to get a better understanding of the disease, which may inspire them to take part in preventing, detecting, controlling and combating the disease. In this paper, we present an explainable data analytics system for disease and healthcare informatics. Our system consists of two key components. The predictor component analyzes and mines historical disease and healthcare data for making predictions on future data. Although huge volumes of disease and healthcare data have been generated, volumes of available data may vary partially due to privacy concerns. So, the predictor makes predictions with different methods. It uses random forest With sufficient data and neural network-based few-shot learning (FSL) with limited data. The explainer component provides the general model reasoning and a meaningful explanation for specific predictions. As a database engineering application, we evaluate our system by applying it to real-life COVID-19 data. Evaluation results show the practicality of our system in explainable data analytics for disease and healthcare informatics.

*Corresponding author: kleung@cs.umanitoba.ca (C.K. Leung)

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-8991-4/21/07...\$15.00
<https://doi.org/10.1145/3472163.3472175>

CCS CONCEPTS

• **Information systems** → **Data analytics**; *Data mining*; • **Applied computing** → **Health care information systems**; **Health informatics**; • **Computing methodologies** → *Supervised learning by classification*; *Classification and regression trees*; • **Mathematics of computing** → *Exploratory data analysis*.

KEYWORDS

Database engineering, database application, data science, disease analytics, disease informatics, health informatics, healthcare informatics, healthcare information system, explainable artificial intelligence (XAI), data mining, prediction, random forest, neural network, few-shot learning (FSL), coronavirus disease 2019 (COVID-19)

ACM Reference Format:

Carson K. Leung, Daryl L.X. Fung, Thanh Huy Daniel Mai, Joglas Souza, Nguyen Duy Thong Tran, and Qi Wen. 2021. Explainable Data Analytics for Disease and Healthcare Informatics. In *25th International Database Engineering & Applications Symposium (IDEAS 2021)*, July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3472163.3472175>

1 INTRODUCTION

With advancements in technology, huge volumes of valuable data have been generated and collected at a rapid velocity from a wide variety of rich data sources. Examples include:

- biodiversity data [1],
- biomedical/healthcare data and disease reports (e.g., COVID-19 statistics) [2–4],
- census data [5],
- imprecise and uncertain data [6–9],
- music data [10, 11],
- patent register [12, 13],
- social networks [14–19],
- time series [20–26],
- transportation and urban data [27–31],
- weather data [32], and
- web data [33–37].

Embedded in these data is implicit, previously unknown and potentially useful information and knowledge that can be discovered by data science [38–41], which make good uses of:

- data mining algorithms [42–50] (e.g., incorporating constraints [51–54]),
- data analytics methods [55–60],
- machine learning techniques [61–63],
- visualization tools [64, 65], and/or
- mathematical and statistical modeling [66].

Analyzing these data can be for social good. For instance, analyzing and mining biomedical/healthcare data and disease helps the discovery of useful information and knowledge about the disease such as:

- severe acute respiratory syndrome (SARS), which was caused by a SARS-associated coronavirus (CoV) and led to an outbreak in 2003.
- Swine flu, which was caused by influenza A virus subtype H1N1 (A/H1N1) and led to an outbreak from 2009 to mid-2010.
- Middle East respiratory syndrome (MERS), which was caused by a MERS-CoV and led to outbreaks in like Middle East (e.g., Saudi Arabia) between 2012–2018 and South Korea in 2015.
- Zika virus disease, which was primarily transmitted by the bite of an infected mosquito and led to an outbreak in Brazil during 2015–2016.
- coronavirus disease 2019 (COVID-19), which was caused by SARS-CoV-2. This was reported to break out in 2019, became a global pandemic in March 2020, and is still prevailing in 2021.

Discovered information and knowledge helps prevent, detect, control and/or combat the disease. This, in turn, helps save patient life and improve quality of our life. Hence, it is useful to have a data analytics system for analyzing and mining these data. In response, we present a data analytics system for analyzing and mining *disease informatics* and *healthcare informatics* (aka *health informatics*).

Take COVID-19 as an example of diseases. Since its declaration as a pandemic, there have been more than 180 million confirmed COVID-19 cases and more than 3.9 million deaths worldwide (as of July 01, 2021)¹. These have led to huge volumes of valuable data. However, partially due to privacy concerns (e.g., to privacy-preserving data publishing) and/or fast reporting of the information, the volume of *available* data may vary. As such, it is important to design the data analytics system in such a way that it could discover useful knowledge based on various volumes of available data. In response, our data analytics system is designed in such a way that it makes predictions on future data based on the analysis and mining on various volumes of historical disease and healthcare data. Specifically, it makes predictions with different methods by using:

- random forest with the available data are *sufficient*, and
- neural network (NN)-based *few-shot learning* (FSL) with the available data are *limited*.

Advancements in technology have led to huge volumes of valuable data—which may be of a wide variety of data types stored in a wide variety of data formats—have been generated and collected at a rapid velocity from a wide variety of rich data sources. They

can also be at different levels of veracity. Hence, in terms of characteristics of these data, they are often characterized by multiple V's (e.g., 5V's, 6V's, 7V's, etc.). These include:

- (1) value, which focuses on data usefulness;
- (2) variety, which focuses on differences in data formats (e.g., comma-separated values (CSV), JavaScript object notation (JSON)), types (e.g., structured relational/transactional data, semi-structured text on the web, unstructured audio or video), and/or sources;
- (3) velocity, which focuses on data generation and collection rates;
- (4) veracity, which focuses on data quality (e.g., precise vs. imprecise and uncertain data);
- (5) volume, which focuses on data quantity; as well as
- (6) validity and visibility, which focus on data interpretation and visualization.

As “a picture is worth a thousand words”, data interpretation and visualization enhance the data validity and visibility. Moreover, for disease and healthcare informatics, having the ability to interpret and visualize the data and/or knowledge discovered by data analytics is desirable. It is because this ability would further enhance user understanding of the disease. Hence, we incorporate *explainable artificial intelligence* (XAI) to our data analytics system so that the resulting explainable data analytics system not only makes accurate predictions but also provides reasoning and meaningful explanations for specific predictions.

In terms of *related works*, there have been works on disease and healthcare informatics. As an example, since the declaration of COVID-19 as a pandemic, researchers have focused on different aspects of the COVID-19 disease. Examples include:

- Some social scientists have studied crisis management for the COVID-19 outbreak [67].
- Some medical and health scientists have focused on clinical and treatment information [68]. Some others have focused on drug discovery and vaccine development (e.g., messenger ribonucleic acid (mRNA) vaccines like Moderna and Pfizer-BioNTech, adenovirus vector vaccines like AstraZeneca and Janssen, inactivated virus vaccines, subunit vaccines) [69].
- Some natural scientists and engineers have examined artificial intelligence (AI)-driven informatics, sensing, imaging for tracking, testing, diagnosis, treatment and prognosis [70] such as those imaging-based diagnosis of COVID-19 using chest computed tomography (CT) images [71, 72]. They have also come up with mathematical modelling of the spread of COVID-19 [73].

Similar to the last category of examples, we also focus on AI-driven informatics—in particular, disease and healthcare informatics. However, unlike the last category, we focus on textual data (e.g., blood test results) instead of images like CT images. Note that medical and healthcare data are expensive to produce. For instance, CT images requires radiologists to supervise the operation of CT scanners, which are often available in large hospitals instead of small clinics. In contrast, blood tests are more regular procedures, which are more accessible and easily carried out by medical staff even in small clinics.

¹<https://covid19.who.int/>

Our *key contributions* of this paper is our explainable data analytics for disease and healthcare informatics. We design the system in such a way that it consists of two key components:

- (1) The predictor component analyzes and mines historical disease and healthcare data for making predictions on future data. It makes predictions with different methods based on the volume of available data. To elaborate, it uses:
 - random forest With sufficient data, and
 - network-based few-shot learning (FSL) with limited data.
- (2) The explainer component provides:
 - the general model reasoning, and
 - a meaningful explanation for specific predictions.

As a database engineering application, we evaluate our system by applying it to real-life blood test results for COVID-19 data. Evaluation results show the practicality of our system in explainable data analytics for disease and healthcare informatics.

The remainder of this paper is organized as follows. The next section discusses related works. Section 3 describes our explainable data analytics for disease and healthcare informatics. Section 4 shows evaluation results and Section 5 draws the conclusions.

2 BACKGROUND AND RELATED WORKS

Recall from Section 1, since its declaration as a pandemic, there have been more than 180 million confirmed COVID-19 cases and more than 3.9 million deaths worldwide (as of July 01, 2021). In Canada, there have been more than 1.4 million confirmed COVID-19 cases and more than 26 thousand deaths². Like many other viruses, SARS-CoV-2 (which has caused COVID-19) also mutates over time. This leads to several SARS-CoV-2 variants³, which can be categorized as:

- *variants of concern (VOC)*:
 - alpha (lineage B.1.1.7), which samples were first documented in the UK in September 2020;
 - beta (B.1.351), which samples were first documented in South Africa in May 2020;
 - gamma (P.1), which samples were first documented in Brazil in November 2020; and
 - delta (lineages B.1.617.2, AY.1, and AY.2), which samples were first documented in India in October 2020.
- *variants of interest (VOI)*:
 - eta (lineage B.1.525), which samples were first documented in December 2020;
 - iota (B.1.526), which samples were first documented in the USA in November 2020;
 - kappa (B.1.617.1), which samples were first documented in India in October 2020;
 - lambda (C.37), which samples were first documented in Peru in December 2020.

Since January 2021, more than 187 thousand Canadian COVID-19 cases have screened and sequenced. As of July 01, 2021, there have

been more than 222 thousand alpha, 2 thousand beta, 18 thousand gamma, and 4 thousand delta cases⁴.

Hence, it is crucial for early detection of COVID-19 infection. To do so, two main types of tests are used:

- antibody tests (aka serology tests), which test for past infection. Specifically, these tests look for the presence of proteins created by the immune system soon after the persons have been infected or vaccinated.
- viral tests (aka diagnostic tests), which test for current infection. Specifically, two common viral tests are:
 - molecular tests—such as polymerase chain reaction (PCR) tests (aka nucleic acid amplification tests (NAATs))—which look for the presence of viral genetic materials, and
 - antigen tests, which look for presence of some specific proteins from the virus.

Between the two main types of viral tests, molecular tests are more costly, more invasive (e.g., require nasopharyngeal (NP) swab) and may take several hours to days in traditional laboratory settings in obtaining the test results when compared with the antigen tests. However, in addition to being specific (in the sense of being able to correctly identify those without the disease, i.e., high true negative rate), molecular tests are more sensitive (in the sense of more capable to correctly identify those with the disease, i.e., higher true positive rate) than the antigen tests. Hence, considering the tradeoff between the two main types of viral tests, it is desirable to have an efficient but also accurate way to determine or predict the results (i.e., positive or negative for diseases like COVID-19).

In data analytics, it is common to train a prediction model with lots of data because data are one of (if not the most important) resources for researchers in many fields. Analysts look for trends, patterns and commonalities within and among samples. However, in the medical and health science field, samples can be expensive to obtain. Moreover, due to privacy concerns and other related issues, few samples may be made available for analyses. This motivates our current work on designing a data analytics system that uses different methods make predictions based on the availability of data (e.g., sufficient or limited data).

In terms of related works, Rustam et al. [74] forecasted new COVID-19 cases in three different dimensions—i.e., number of new cases, death rate, and recovery rate—based on historical time series on these dimensions collected by John Hopkins University. Although their model was useful in forecasting new cases, it relied on how extensively the population has been tested. Data from regions without extensive tests (e.g., due to lack of resources) could lead to unreliable predictions. In contrast, our current work focuses on forecasting the positive and negative COVID-19 cases.

Brinati et al. [75] presented a machine learning solution for predicting COVID-19 with routine blood test data collected from a hospital in Milan, Italy. They analyzed a dataset of 279 patients, of which 177 were tested positive and 102 were tested negative. They observed that the random forest (RF) model led to an accurate prediction. As for the model interpretation (e.g., level of importance

²<https://www.ctvnews.ca/health/coronavirus/tracking-every-case-of-covid-19-in-canada-1.4852102>

³<https://www.who.int/en/activities/tracking-SARS-CoV-2-variants/>

⁴<https://www.ctvnews.ca/health/coronavirus/tracking-variants-of-the-novel-coronavirus-in-canada-1.5296141>

of the features), they relied on validation from solely healthcare professional. Like theirs, our current work also uses RF when the available data are sufficient. However, unlike theirs, our current work provides healthcare professional with the general reasoning of the prediction model and meaningful explanations on specific predictions. This helps reduce the workload of busy healthcare professional.

Xu et al. [76] combined the lab testing data and CT scan images to develop a robust model, which helped differentiate COVID-19 cases from other similar diseases and the degree of severity. Their model classified the patients according to the following labels: non-severe or severe COVID-19, as well as healthy or viral pneumonia. They used a combination of traditional machine learning models (e.g., RF) and deep learning (e.g., convolutional neural networks (CNN)). However, it is important to note that CT scan images are expensive to produce. Moreover, sophisticated deep learning techniques may also incur high computational costs. In contrary, our current work does not rely on the expensive CT scan images or computationally intensive deep learning techniques, while maintaining reasonably well prediction results.

While the RF can handle situations in which data are sufficient, we explore *few-shot learning* (FSL) [77] to handle situations in which data are limited. In general, FSL is a machine learning technique, which aims to learn from a limited number of examples in experience with supervised information for some classes of task. It has become popular. For instance, Snell et al. [78] applied the FSL to the miniImageNet dataset with 5-shot modelling and to the Omniglot dataset.

3 OUR EXPLAINABLE DATA ANALYTICS SYSTEM

3.1 Overview

Our explainable data analytics system for disease and healthcare informatics consists of two key components:

- (1) predictor component, which analyzes and mines historical disease and healthcare data for making predictions on future data. To handle different levels of data availability, it uses:
 - the random forest (RF) when data are sufficient, and
 - the neural network (NN)-based few-shot learning (FSL) when data are limited.
- (2) explainer component, which provides:
 - the general model reasoning, and
 - meaningful explanations to specific predictions.

3.2 Our Predictor for Sufficient Data

To aim for accurate prediction, our predictor first cleans data and engineers features. Although the imputation techniques (which usually serve to preserve observations on missing information) work well in many applications, we choose not to use them for medical data because any changes in the range of clinical values could lead to misleading results. Instead, given a dataset with many variables (aka features or columns) and observations, we clean the data by first profiling them. We preserve all columns with less than certain threshold (say, 90%) of missing values. To reduce

dimensionality, we remove columns that are highly correlated to others because such a removal would lead to the same outcome due to their high correlation. Then, we remove columns with constant values because they do not offer any nuances between observations.

Afterwards, our predictor applies some engineering to the remaining features. As an example for COVID-19 data, when the number of records with missing values is high, we create an extra column to detect if the patient was tested positive for any other viruses (say, 19 other viruses). This single column helps reduce the sparsity of many columns (e.g., these 19 columns). We then transform categorical features to numerical ones, and then use the Pearson's Correlation test—as shown in Eq. (1)—to check the feature interactions with the target outcome.

$$\rho_{X,Y} = \frac{\text{cov}(X, Y)}{\sigma_X \sigma_Y} \quad (1)$$

where

- $\text{cov}(X, Y)$ is the covariance of X and Y ; and
- σ_X and σ_Y are standard deviations of X and Y , respectively.

Once the data are cleaned and engineered, our predictor splits the dataset into training and testing sets. It also applies stratification when doing so. In other words, the distribution of each class (positive and negative) was proportional between the two sets. The difference in the ratio between positive and negative classes in the training set may cause the “class imbalance” problem, in which one class dominates the learning process of the model. To avoid this problem, we apply a k -means clustering on the training set to reduce the number of negative classes. By doing so, we eliminate some of the instances based on clustering similarities and removing data points similar to others that remained. We also explore a good ratio.

In addition, our predictor applies a random search on the training dataset to find the best set of hyperparameters for RF prediction. As an ensemble algorithm for machine learning classification and regression prediction, RF evolves from the decision tree algorithm. It enhances the predictive power by using a forest of multiple trees. Each tree receives different portions of the dataset, and the final prediction result is an average of the individual's decision tree results available in the forest. To evaluate the performance and determine the best set of hyperparameters, we apply a 10-fold cross-validation to evaluate the results in different portions of the dataset. The cross-validation technique serves to approximate the model results of an unseen dataset, and thus avoids over-fitting.

3.3 Our Predictor for Limited Data

To handle situations in which the available data are limited, our predictor uses FSL. With an autoencoder architecture, it learns internal representations by error propagation. Specifically, once the data are cleaned and engineered (as described in Section 3.2), it first builds an *autoencoder* with four fully connected layers to reconstruct the input features. Here, the input layer takes n features. With a hidden layer, the encoder module of the autoencoder maps the input features $\{F_i\}_{i=1}^n$ to produce encoded features. With another hidden layer, the decoder module of the autoencoder maps the encoded features to produce reconstructed input $\{R_i\}_{i=1}^n$. During the process, the autoencoder aims to minimize the loss function

L_R in the reconstruction of input features. This loss function L_R is computed as a mean absolute error:

$$L_R = \frac{1}{n} \sum_{i=1}^n |F_i - R_i| \quad (2)$$

where

- n is the number of trained data samples,
- F_i is one of the n features $\{F_i\}_{i=1}^n$ fed into (the encoder module of) the autoencoder, and
- R_i is one of the n reconstructed features $\{R_i\}_{i=1}^n$ reconstructed by (the decoder module of) the autoencoder.

Afterward, the predictor *freezes* the autoencoder and builds a neural network with two fully connected layers to make prediction on class label (i.e., positive or negative). Here, the input layer takes encoded features from the frozen autoencoder (specifically, the encoder module). With a hidden layer, the prediction module maps the encoded features to produce a single-label prediction:

$$y_i = \begin{cases} 0 & \text{for negative class label} \\ 1 & \text{for positive class label} \end{cases} \quad (3)$$

During the process, the autoencoder aims to minimize the loss function L_P in the reconstruction of input features. This loss function L_P is computed as a binary cross-entropy loss:

$$L_P = \frac{1}{m} \sum_{i=1}^m [y_i \log(y_i) + (1 - y_i) \log(1 - y_i)] \quad (4)$$

where m is the total number of limited data samples per class label.

3.4 Our Explainer

To provide explanations to user, once the data are cleaned and engineered, our explainer first computes Pearson’s correlation as per Eq. (1) to find correlation among different cleaned and engineered features. It represents the correlation in a heat map.

In addition, our explainer also produce a partial dependence plot (PDP) explaining the interactions between independent features and the target one. The PDP also explains combination of feature importance.

Following the general explanation given above, our explainer also explores specific instances of explanations to have a deeper understanding level. To elaborate, for each specific prediction, our explainer examines the degree of positive or negative contribution of each feature by showing a bar chart. So, features are listed in the y -axis, and degrees of (positive or negative) contribution are indicated in the x -axis. The range of degree of positive contributions goes from 0 to +1, whereas the range of degree of negative contributions goes from 0 to −1. Values of each attribute are normalized with the average of being 0, maximum values are normalized to +1 and minimum values are normalized to −1. The maximum and minimum degrees indicate a high positive and negative contribution, respectively.

4 EVALUATION

To evaluate our system, we conducted experiments on a real-life open dataset on COVID-19 dataset⁵. Specifically, the dataset contains samples collected to perform the SARS-CoV-2 reverse transcription polymerase chain reaction (RT-PCR) and additional laboratory tests during a visit to a hospital in the state of Sao Paulo, Brazil. After data cleaning and preprocessing, the dataset contains 5,644 potential patients (COVID-19 or not), aka samples or instances. Each instance captures:

- an ID;
- age group;
- hospitalization status—such as (a) admitted to regular ward, (b) admitted to semi-intensive unit (SIU), or (c) admitted to intensive care unit (ICU) ward);
- 106 features (69 numerical features and 37 categorical features); and
- a class label (i.e., tested positive or negative).

In term of data distribution, 558 instances (i.e., 9.9% of 5,644 instances in the dataset) were tested/labelled “positive” for the SARS-CoV-2 test result, and the remaining 5,086 instances (i.e., 90.1%) were tested/labelled “negative”.

Initially, we started with a dataset with 112 variables (aka features or columns) and 5,644 observations. Observing that more than 88% of the values were missing, we preserved columns with less than 90% of missing values. Then, we removed highly correlated columns and constant-value columns. We also observed that 19 variables were there to indicate the presence of other viruses (e.g., adenovirus, influenza A, rhinovirus). As such, to further reduce dimensionality, we replace these 19 columns by a single column to indicate the instance was tested positive for at least one of the 19 viruses. At the end of the data cleaning and feature engineering process, we were left with 16 important features.

Figure 1 shows a heat map with all the computed correlations. All rows except the last one (i.e., first 15 rows) are independent variables, and the last row corresponds to the target feature “SARS-CoV2 exam result”. Observed from this figure, both leukocytes and platelets are features that present a higher correlation with the target variable. From immunological and clinical viewpoints, *leukocyte* is a general term for all white blood cell, and *platelet* is a type of white blood cells in blood in charge wound healing. Thus, they are highly correlated. Based on that, we eliminated any rows that had missing values for these two variables. We ended up with a dataset with (16 features and) 598 rows (i.e., observations).

We split the dataset with stratification into 70% for training and 30% for testing such that each class (positive and negative) was proportional between the two sets. Consequently, we used 418 observations (with 361 negatives and 57 positives) for training set, and 180 instances (with 156 negatives and 24 positives) for testing. Observing the ratio between positive and negative classes in the training set was around 6.3 (i.e., “class imbalance”), we applied a k -means clustering on the training set to reduce the number of negative classes by eliminating some instances based on clustering similarities and removing similar data points in the remainder.

⁵<https://www.kaggle.com/einsteindata4u/covid19>

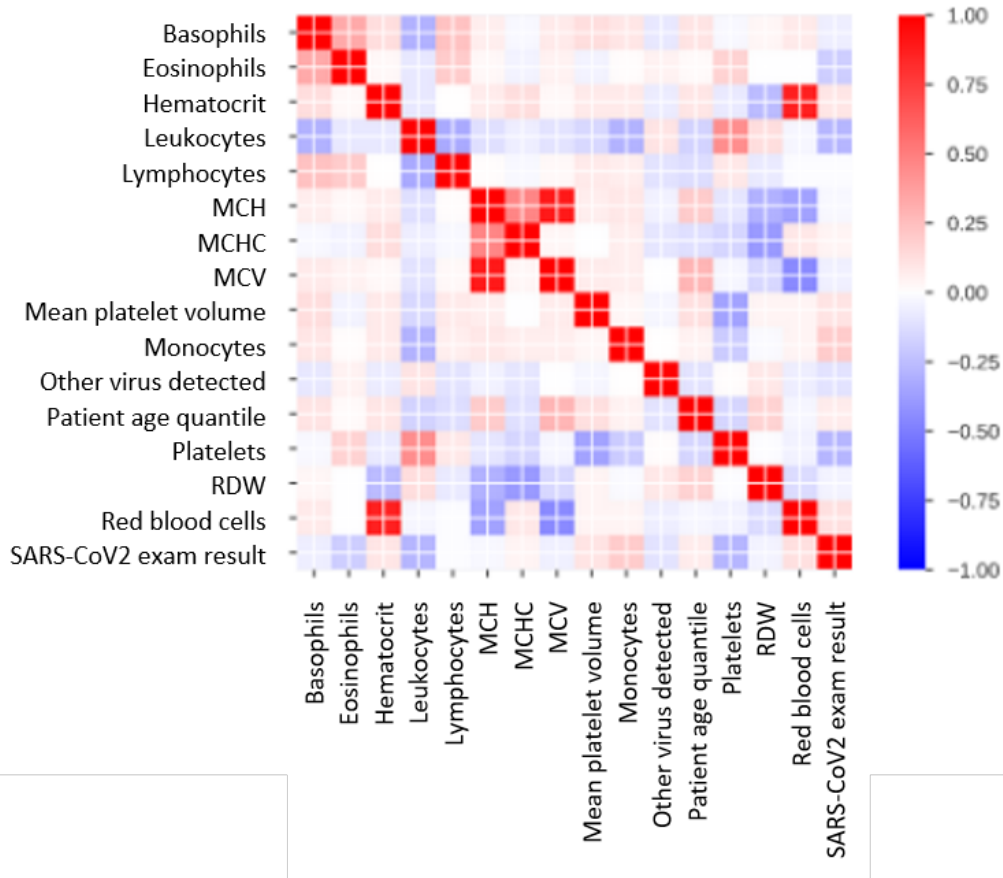


Figure 1: Our explainer shows a heat map to explain correlations among dependent and target features

Consequently, we reduce the ratio from 6.3 to 1.52 (with 87 negatives and 57 positives).

We ran our RF-based predictor with these 418 training observations and 180 testing instances. In addition, we also compared our predictor with several existing ML models:

- logistic regression [79],
- decision tree [79],
- gradient boosting [79], and
- linear support vector classification (SVC) [79].

We measured F1 score, which is computed by:

$$\text{F1 score} = \frac{2TP}{2TP + FP + FN} \quad (5)$$

where

- TP is true positives,
- FP is false positives, and
- FN is false negatives.

Each experiment was ran for 5-fold cross validation and the average F1 score was computed. Table 1 shows that our RF-based predictor makes accurate predictions with sufficient data. It led to the highest F1 score, the highest number of true positives (TP), and the lowest

Table 1: Comparison of our random forest (RF)-based predictor with related works

	F1	TP	FN
Logistic regression [79]	0.86	45.6	3.0
Decision tree [79]	0.81	42.2	6.4
Gradient boosting [79]	0.85	44.0	4.6
Linear SVC [79]	0.85	44.8	3.8
Our RF-based predictor	0.86	46.2	2.4

number of false negatives (FN). Note that FN may lead to late detection of untreated patients, causing wider spread of the disease, which in turn may lead to more cases.

Moreover, we also ran our NN-based FSL predictor with a few samples (e.g., 1 and 5 samples for each class label) from the 418 training observations and 180 testing instances. Again, we also compared our predictor with the aforementioned existing ML models. Tables 2 and 3 both show that our NN-based FSL predictor makes accurate predictions with limited data. It led to the highest F1 score, the highest number of true positives (TP), and the lowest number of

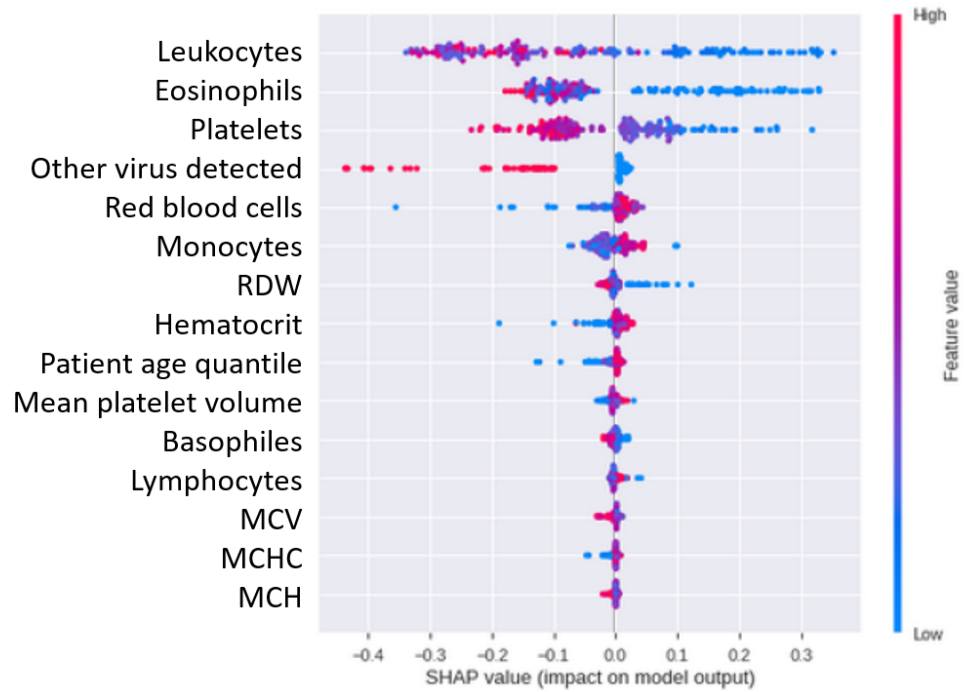


Figure 2: Our explainer shows a partial dependence plot to explain feature importance

Table 2: Comparison of our neural network (NN)-based few-shot learning (FSL) predictor with related works when using 1 sample per class label

1 sample per class	F1	TP	FN
Logistic regression [79]	0.52	108.6	133.4
Decision tree [79]	0.52	127.0	115.0
Gradient boosting [79]	0.55	112.4	129.6
Linear SVC [79]	0.55	123.2	118.8
Our NN-based FSL predictor	0.75	241.0	1.0

Table 3: Comparison of our NN-based FSL predictor with related works when using 5 samples per class label

5 samples per class	F1	TP	FN
Logistic regression [79]	0.68	156.4	81.6
Decision tree [79]	0.66	152.2	85.8
Gradient boosting [79]	0.66	153.2	84.8
Linear SVC [79]	0.63	143.4	94.6
Our NN-based FSL predictor	0.70	225.2	12.8

false negatives (FN). Note that FN may lead to late detection of untreated patients, causing wider spread of the disease, which in turn may lead to more cases.

The above evaluation results show the effectiveness and practicality of our predictors (with sufficient and limited data). In addition,

we also evaluated our explainer. For instance, it captures some insights from the model reason for the predictions by producing a partial dependence plot (PDP)—as shown in Figure 2—to explain the interactions between independent features and the target one. An interpretation of this figure is as follows:

- The most important feature for the prediction is “Leukocytes”. In the immunological and clinical context, leukocyte is a general term for all white blood cell.
- The least important is “Mean corpuscular hemoglobin”. In the immunological and clinical context, mean corpuscular hemoglobin concentration (MCHC) measures the concentration of haemoglobin in a given volume of packed red blood cell. It can be computed by dividing the hemoglobin by the hematocrit. A low level of MCHC is an indication of anemia.
- Low levels of leukocytes, eosinophils and platelets all increase the probability towards a positive prediction (COVID-19 positive). Again, in the immunological and clinical context, eosinophils is a type of disease-fighting white blood cell. A high level of eosinophils often indicates a parasitic infection, an allergic reaction or cancer.
- When other kinds of viruses are detected in the patient, the probability increases towards a negative prediction (COVID-19 negative).

Following the general explanation given above, our explainer also explores specific instances of explanations to have a deeper understanding level. To elaborate, for each specific prediction, our explainer examines the degree of positive or negative contribution of each feature by showing a bar chart. See Figures 3 and 4 for two examples.

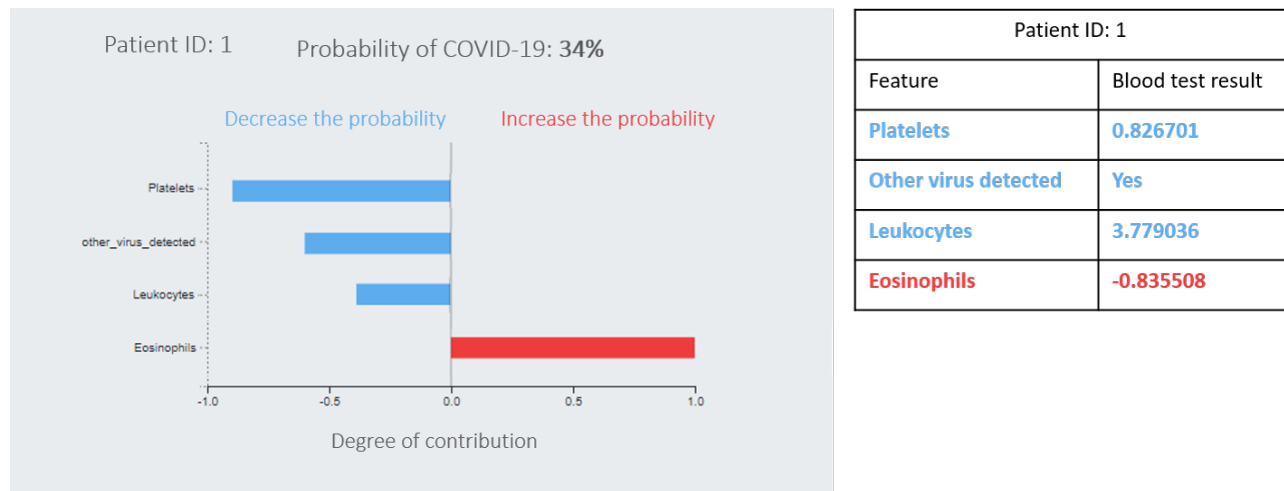


Figure 3: Our explainer provides an explanation for a COVID-19 true negative instance

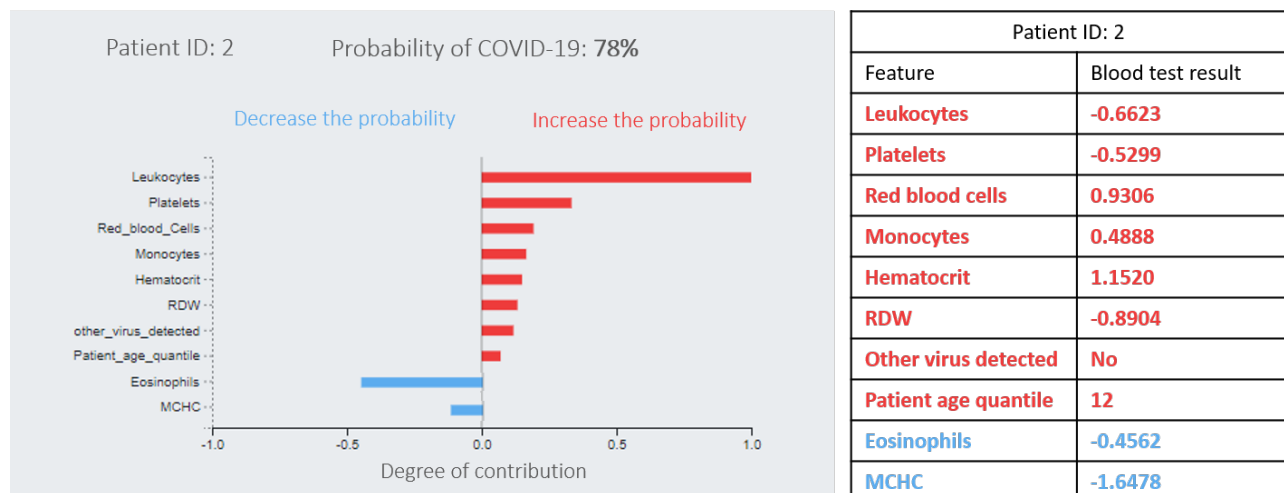


Figure 4: Our explainer provides an explanation for a COVID-19 true positive instance

To elaborate, Figure 3, our predictor makes a prediction that the patient had 34% of chances to have the COVID-19, i.e., likely to be negative in our binary classification model. Our explainers provide the specific reasoning for this instance: “Platelets,” “other virus detected” and “leukocytes” contribute negatively because their (negative) values indicate these features for this specific patient decreased the probability of being a positive diagnostic, i.e., increased the probability of being a negative diagnostic. We observe that these negatively contributing features follow the general explanations provided by our explainer in Figure 2. Hence, our explainer supports clinicians in trusting the prediction made by our predictor.

Moreover, these observations and explanations—e.g., high level of platelets, detection of other virus, high level of leukocytes (i.e., high number of white blood cells)—are consistent with the medical literature. To elaborate, platelets are involved in blood clotting. It was evidenced that “blood tests in symptomatic COVID-19 show a

deficiency of lymphocytes and white blood cells, in general. Lower platelet counts are a marker of higher mortality in hospitalized patients.” [80] “Early reports from China suggested that co-infection with other respiratory pathogens was rare. If this were the case, patients positive for other pathogens might be assumed unlikely to have SARS-CoV-2. The Centers for Disease Control and Prevention endorsed testing for other respiratory pathogens, suggesting that evidence of another infection could aid the evaluation of patients with potential COVID-19 in the absence of widely available rapid testing for SARS-CoV-2.” [81, 82]. Hence, high levels of platelets and leukocytes, as well as detection of other virus, are key contributing factors to true negatives.

Figure 4 shows an example of our explainer in providing explanation for a predicted COVID-19 positive diagnostic with 78% probability. Here, we had many factors that increased the probability of a confirmed diagnosis. Again, we observe that these positively

contributing features follow the general explanations previously described in Figure 2. Moreover, these observations and explanations—e.g., low level of leukocytes (aka leukopenia, which refers to a low level of white blood cells), low level of platelet—are consistent with the medical literature. It was evidenced that “blood tests in symptomatic COVID-19 show a deficiency of lymphocytes and white blood cells, in general. Lower platelet counts are a marker of higher mortality in hospitalized patients.” [80] Hence, low levels of leukocytes and platelets, together with some other features, are key contributing factors to true positives.

5 CONCLUSIONS

In this paper, we present an explainable data analytics system for disease and healthcare informatics. Our system consists of two key components. The predictor component analyzes and mines historical disease and healthcare data for making predictions on future data. As volumes of available data may vary, the predictor makes predictions with a random forest model when the available data are sufficient. It makes predictions with a neural network-based few-shot learning model when the available data are limited. The explainer component provides the general model reasoning for the prediction by showing heat maps to explain correlations among dependent features and target class labels, as well as partial dependence plot to explain importance of these features. In addition to general model reasoning, our explainer also provides explanations to specific instance by showing how the normalized values of features increase or decrease the probability of having a positive label. In other words, it explains the feature values that contributing towards a positive or negative prediction made by our predictor. As a database engineering application, we demonstrate and evaluate our system by applying it to real-life COVID-19 data. Evaluation results show the practicality of our system in explainable data analytics for disease and healthcare informatics on COVID-19. It is important to note that our system is designed in such a way that it is capable of handling other disease. As *ongoing and future work*, we transfer knowledge learned here to disease and healthcare informatics of other disease. We also explore incorporating user preference in providing users with explainable data analytics for disease and healthcare informatics.

ACKNOWLEDGMENTS

This work is partially supported by Natural Sciences and Engineering Research Council of Canada (NSERC), as well as the University of Manitoba.

REFERENCES

- [1] I.M. Anderson-Grégoire, et al. 2021. A big data science solution for analytics on moving objects. In AINA 2021, vol. 2, 133-145.
- [2] A. Demiraj, et al. 2019. Meta-data management and quality control for the medical informatics platform. In IDEAS 2019, 10:1-10:9.
- [3] H.H. Le, et al. 2019. Differentially private sequential pattern mining considering time interval for electronic medical record systems. In IDEAS 2019, 13:1-13:9.
- [4] S. Shang, et al. 2020. Spatial data science of COVID-19 data. In IEEE HPCC-SmartCity-DSS 2020, 1370-1375.
- [5] C.M. Choy, et al. 2021. Natural sciences meet social sciences: census data analytics for detecting home language shifts. In IMCOM 2021, 520-527. DOI: 10.1109/IMCOM51814.2021.9377412
- [6] A.D. Borhanian, F. Sadri. 2013. A compact representation for efficient uncertain-information integration. In IDEAS 2013, 122-131.
- [7] C.K. Leung, D.A. Braszczuk. 2009. Mining uncertain data for constrained frequent sets. In IDEAS 2009, 109-120.
- [8] C.K. Leung, D.A. Braszczuk. 2010. uCSF2: an enhanced system that mines uncertain data for constrained frequent sets. In IDEAS 2010, 32-37.
- [9] J. Zheng, et al. 2015. Asymptotic-efficient algorithms for skyline query processing over uncertain contexts. In IDEAS 2015, 106-115.
- [10] K.E. Barkwell, et al. 2018. Big data visualisation and visual analytics for music data mining. In IV 2018, 235-240.
- [11] N. Kito, et al. 2015. Correlation analysis among the metadata-based similarity, acoustic-based distance, and serendipity of music. In IDEAS 2015, 198-199.
- [12] W. Lee, et al. 2014. Reducing noises for recall-oriented patent retrieval. In IEEE BDCloud 2014, 579-586.
- [13] C.K. Leung, et al. 2019. Information technology-based patent retrieval model. In Springer Handbook of Science and Technology Indicators, 859-874.
- [14] C. Aarts, et al. 2020. A practical application for sentiment analysis on social media textual data. In IDEAS 2020, 26:1-26:6.
- [15] I. Afyouni, et al. 2020. Spatio-temporal event discovery in the big social data era. In IDEAS 2020, 7:1-7:6.
- [16] G. Bergami, et al. 2019. On approximate nesting of multiple social network graphs: a preliminary study. In IDEAS 2019, 40:1-40:5.
- [17] D. Choudhery, C.K. Leung. 2017. Social media mining: prediction of box office revenue. In IDEAS 2017, 20-29.
- [18] F. Jiang, et al. 2012. Finding popular friends in social networks. In CGC 2012, 501-508.
- [19] S.P. Singh, et al. 2019. A theoretical approach to discover mutual friendships from social graph networks. In iiWAS 2019, 212-221.
- [20] A.K. Chanda, et al. 2017. A new framework for mining weighted periodic patterns in time series databases. ESWA 79, 207-224.
- [21] C.K. Leung, et al. 2008. Efficient algorithms for stream mining of constrained frequent patterns in a limited memory environment. In IDEAS 2008, 189-198.
- [22] C.K. Leung, et al. 2011. A landmark-model based system for mining frequent patterns from uncertain data streams. In IDEAS 2011, 249-250.
- [23] C.K. Leung, Q.I. Khan. 2006. Efficient mining of constrained frequent patterns from streams. In IDEAS 2006, 61-68.
- [24] K.J. Morris, et al. 2018. Token-based adaptive time-series prediction by ensembling linear and non-linear estimators: a machine learning approach for predictive analytics on big stock data. IEEE ICMLA 2018, 1486-1491.
- [25] M. Orang, N. Shiri. 2014. An experimental evaluation of similarity measures for uncertain time series. In IDEAS 2014, 261-264.
- [26] K.S. Perera, et al. 2016. Efficient approximate OLAP querying over time series. In IDEAS 2016, 205-211.
- [27] A.A. Audu, et al. 2019. An intelligent predictive analytics system for transportation analytics on open data towards the development of a smart city. In CISIS 2019, 224-236.
- [28] P.P.F. Balbin, et al. 2020. Predictive analytics on open big data for supporting smart transportation services. Procedia Computer Science 176, 3009-3018.
- [29] J. Chabin, et al. 2018. A context-driven querying system for urban graph analysis. In IDEAS 2018, 297-301.
- [30] C.K. Leung, et al. 2018. Effective classification of ground transportation modes for urban data mining in smart cities. In DaWaK 2018, 83-97.
- [31] C.K. Leung, et al. 2019. Urban analytics of big transportation data for supporting smart cities. In DaWaK 2019, 24-33.
- [32] T.S. Cox, et al. 2018. An accurate model for hurricane trajectory prediction. In IEEE COMPSAC 2018, vol. 2, 534-539.
- [33] B.C. Desai, et al. 2020. The web: a hacker's heaven and an on-line system. In IDEAS 2020, 6:1-6:7.
- [34] M. Endres, et al. 2020. Lifting preferences to the semantic web: PreferenceSPARQL. In IDEAS 2020, 19:1-19:8.
- [35] C.K. Leung, et al. 2018. Web page recommendation from sparse big web data. In IEEE/WIC/ACM WI 2018, 592-597.
- [36] C.K. Leung, et al. 2020. Explainable machine learning and mining of influential patterns from sparse web. In IEEE/WIC/ACM WI-IAT 2020, 829-836.
- [37] S.P. Singh, et al. 2020. Analytics of similar-sounding names from the web with phonetic based clustering. In IEEE/WIC/ACM WI-IAT 2020, 580-585.
- [38] V. Charles, et al. 2021. Special issue data science for better productivity. J. Oper. Res. Soc. 72(5), 971-974.
- [39] C.K. Leung, et al. 2020. Data science for healthcare predictive analytics. In IDEAS 2020, 8:1-8:10.
- [40] C.K. Leung, F. Jiang. 2014. A data science solution for mining interesting patterns from uncertain big data. In IEEE BDCloud 2014, 235-242.
- [41] I. Martinez, et al. 2021. Data science methodologies: current challenges and future approaches. Big Data Res. 24, 100183:1-100183:18.
- [42] M.T. Alam, et al. 2021. Mining frequent patterns from hypergraph databases. In PAKDD 2021, Part II, 3-15.
- [43] M.T. Alam, et al. 2021. Discriminating frequent pattern based supervised graph embedding for classification. In PAKDD 2021, Part II, 16-28.
- [44] S. Daggumati, P.Z. Revesz. 2019. Data mining ancient scripts to investigate their relationships and origins. In IDEAS 2019, 26:1-26:10.

- [45] A. Fariha, et al. 2013. Mining frequent patterns from human interactions in meetings using directed acyclic graphs. In PAKDD 2013, Part I, 38–49.
- [46] W. Lee, et al. (eds.). 2021. Big Data Analyses, Services, and Smart Data.
- [47] C.K. Leung, et al. 2012. Mining probabilistic datasets vertically. In IDEAS 2012, 199–204.
- [48] C.K. Leung. 2014. Uncertain frequent pattern mining. In Frequent Pattern Mining, 417–453.
- [49] C.K. Leung. 2019. Pattern mining for knowledge discovery. In IDEAS 2019, 34:1–34:5.
- [50] K.K. Roy, et al. 2021. Mining sequential patterns in uncertain databases using hierarchical index structure. In PAKDD 2021, Part II, 29–41.
- [51] M. Calautti, et al. 2020. Consistent query answering with prioritized active integrity constraints. In IDEAS 2020, 3:1–3:10.
- [52] C.K. Leung. 2004. Interactive constrained frequent-pattern mining system. In IDEAS 2004, 49–58.
- [53] C.K. Leung, et al. 2012. A constrained frequent pattern mining system for handling aggregate constraints. In IDEAS 2012, 14–23.
- [54] P.Z. Revesz, Y. Li. 1997. MLPQ: a linear constraint database system with aggregate operators. In IDEAS 1997, 132–137.
- [55] A. Gillet, et al. 2020. Empowering big data analytics with polystore and strongly typed functional queries. In IDEAS 2020, 13:1–13:10.
- [56] F. Jiang, C.K. Leung. 2015. A data analytic algorithm for managing, querying, and processing uncertain big data in cloud environments. Algorithms 8(4), 1175–1194.
- [57] C.K. Leung. 2018. Big data analysis and mining. In Encyclopedia of Information Science and Technology, 4e, 338–348.
- [58] C.K. Leung, et al. 2007. An effective multi-layer model for controlling the quality of data. In IDEAS 2007, 28–36.
- [59] C.K. Leung, F. Jiang. 2015. Big data analytics of social networks for the discovery of “following” patterns. In DaWaK 2015, 123–135.
- [60] R. McClatchey, et al. 2016. Provenance support for biomedical big data analytics. In IDEAS 2016, 386–391.
- [61] S. Ahn, et al. 2019. A fuzzy logic based machine learning tool for supporting big data business analytics in complex artificial intelligence environments. In FUZZ-IEEE 2019, 1259–1264.
- [62] C.K. Leung, et al. 2014. A machine learning approach for stock price prediction. In IDEAS 2014, 274–277.
- [63] Z. Sadri, et al. 2020. DRLindex: deep reinforcement learning index advisor for a cluster database. In IDEAS 2020, 11:1–11:8.
- [64] C.K. Leung, C.L. Carmichael. 2009. FpVAT: A visual analytic tool for supporting frequent pattern mining. ACM SIGKDD Explorations 11(2), 39–48.
- [65] Y. Seong, et al. 2020. Guidelines for cybersecurity visualization design. In IDEAS 2020, 25:1–25:6.
- [66] C.K. Leung. 2018. Mathematical model for propagation of influence in a social network. In Encyclopedia of Social Network Analysis and Mining, 2e, 1261–1269.
- [67] W. Kuo, J. He. 2020. Guest editorial: crisis management - from nuclear accidents to outbreaks of COVID-19 and infectious diseases. IEEE Trans. Reliab. 69(3), 846–850.
- [68] A.A. Ardakani, et al. 2020. Application of deep learning technique to manage COVID-19 in routine clinical practice using CT images: results of 10 convolutional neural networks. Comp. Bio. Med. 121, 103795:1–103795:9.
- [69] B. Robson. 2020. COVID-19 coronavirus spike protein analysis for synthetic vaccines, a peptidomimetic antagonist, and therapeutic drugs, and analysis of a proposed achilles’ heel conserved region to minimize probability of escape mutations and drug resistance. Comp. Bio. Med. 121, 103749:1–103749:28.
- [70] A.A. Amini, et al., Editorial special issue on “AI-driven informatics, sensing, imaging and big data analytics for fighting the COVID-19 pandemic”. IEEE JBHI 24(10), 2020, pp. 2731–2732.
- [71] Q. Liu, et al. 2020. A two-dimensional sparse matrix profile DenseNet for COVID-19 diagnosis using chest CT images. IEEE Access 8, 213718–213728.
- [72] D. Shen, et al. 2020. Guest editorial: special issue on imaging-based diagnosis of COVID-19. IEEE TMI 39(8), 2569–2571.
- [73] A. Viguerie, et al. 2021. Simulating the spread of COVID-19 via a spatially-resolved susceptible-exposed-infected-recovered-deceased (SEIRD) model with heterogeneous diffusion. Appl. Math. Lett. 111, 106617:1–106617:9.
- [74] F. Rustam, et al. 2020. COVID-19 future forecasting using supervised machine learning models. IEEE Access 8, 101489–101499.
- [75] D. Brinati, et al. 2020. Detection of COVID-19 infection from routine blood exams with machine learning: a feasibility study. Journal of Medical Systems 44(8), 135:1–135:12.
- [76] M. Xu, et al. 2021. Accurately differentiating between patients with COVID-19, patients with other viral infections, and healthy individuals: multimodal late fusion learning approach. J. Med. Internet Res. 23(1), e25535:1–e25535:17.
- [77] Y. Wang, et al. 2020. Generalizing from a few examples: a survey on few-shot learning. ACM CSur 53(3), 61:1–63:34.
- [78] J. Snell, et al. 2017. Prototypical networks for few-shot learning. In NIPS 2017, 4077–4087.
- [79] C. Sammut, G.I. Webb. 2017. Encyclopedia of Machine Learning and Data Mining.
- [80] L. Thomas. 2020. Platelets may be contributing to COVID-19. News Medical. <https://www.news-medical.net/news/20200624/Platelets-may-be-contributing-to-COVID-19.aspx>
- [81] D. Kim, et al. 2020. Rates of co-infection between SARS-CoV-2 and other respiratory pathogens. JAMA 323(20), 2085–2086.
- [82] N. Chen, et al. 2020. Epidemiological and clinical characteristics of 99 cases of 2019 novel coronavirus pneumonia in Wuhan, China: a descriptive study. Lancet. 395(10223), 507–513.

ICIX: A Semantic Information Extraction Architecture

Angel L. Garrido
garrido@unizar.es
University of Zaragoza
Zaragoza, Spain
Polytechnic University of Valencia
Valencia, Spain

Alvaro Peiro
apeiro@isyc.com
InSynergy Consulting S.A.
Madrid, Spain

Cristian Roman
croman@isyc.com
InSynergy Consulting S.A.
Madrid, Spain

Carlos Bobed
cbobed@unizar.es
University of Zaragoza
Zaragoza, Spain

Eduardo Mena
emena@unizar.es
University of Zaragoza
Zaragoza, Spain

ABSTRACT

Public and private organizations produce and store huge amounts of documents which contain information about their domains in non-structured formats. Although from the final user's point of view we can rely on different retrieval tools to access such data, the progressive structuring of such documents has important benefits for daily operations. While there exist many approaches to extract information in open domains, we lack tools flexible enough to adapt themselves to the particularities of different domains.

In this paper, we present the design and implementation of ICIX, an architecture to extract structured information from text documents. ICIX aims at obtaining specific information within a given domain, defined by means of an ontology which guides the extraction process. Besides, to optimize such an extraction, ICIX relies on document classification and data curation adapted to the particular domain. Our proposal has been implemented and evaluated in the specific context of managing legal documents, with promising results.

CCS CONCEPTS

• **Information systems** → **Document structure**; *Ontologies*; *Content analysis and feature selection*.

KEYWORDS

Information extraction, ontologies, text classification

ACM Reference Format:

Angel L. Garrido, Alvaro Peiro, Cristian Roman, Carlos Bobed, and Eduardo Mena. 2021. ICIX: A Semantic Information Extraction Architecture. In *25th International Database Engineering*

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 978-1-4503-8991-4/21/07...\$15.00

<https://doi.org/10.1145/3472163.3472174>

E Applications Symposium (IDEAS 2021), July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 9 pages.
<https://doi.org/10.1145/3472163.3472174>

1 INTRODUCTION

Information Extraction (IE) is the task of automatically extracting structured information from unstructured or semi-structured documents which, in most cases, involves processing human language texts. IE is pervasive in all kinds of fields (e.g., science, laws, news, etc.), and it is still being done manually in many organizations dedicated to document management. These manually IE tasks are very time-consuming, pricey activities, and subject to many human errors. In recent years, the use of automatic IE tools has gradually gained popularity thanks to the good results obtained in tasks related to obtaining structured data from text-based documents [14]. These techniques provide user with benefits which contribute to their adoption in all types of organizations, private and public, as well as to the emergence of dedicated software and companies that offer this type of services. Even so, when dealing with IE, we face one of the great problems related to Artificial Intelligence and Natural Language Processing (NLP): After almost forty years, the algorithms created to solve this type of issues are still far from being generic enough to operate with any natural language and type of document. In fact, when focusing on IE that achieves optimal results, one of the greatest difficulties in developing a generic IE system is the strong dependence on 1) the domain of the handled documents and 2) the specific target natural language [1].

Apart from the domain generalization difficulties, we have also to bear in mind both document pre and post-processing. Regarding pre-processing, past and current NLP applications usually work with well-formed texts, but when dealing with real scenarios (e.g., health, legal, or industrial documents), it is usual that the owner of the documents must keep the original ones, and therefore, NLP systems have to work just with digital copies (i.e., scanned versions of the documents). So, in these situations, it is mandatory to apply an Optical Character Recognition system (OCR from now on) to identify the text of the document before analyzing it. This introduces noise derived from errors in the recognition process which compromises the accuracy of the information extraction task.

Its automatic correction also implies the adaptation of the algorithms to the working context. Regarding post-processing, automatic information extraction systems usually do not have means to check the quality of data, searching for possible errors (a task that it is heavily domain-dependent). Among the most common IE errors, we can mention: empty or duplicate data, poorly structured data, data distributed among several entities which should refer to just one entity and vice versa [24]. As a result, it is very complex to create a system capable of correctly solving these IE specific tasks in a general way for any domain / use case [28]. Under these circumstances, research efforts that minimize the customization requirements are always welcome, as they contribute to facilitating the design of domain-adaptable systems.

In this paper, we present our approach to deal with these scenarios where we need to extract specific data from extensive documents belonging to a particular domain. In particular, we focus on the following objectives: 1) to create a generic architecture applicable to multiple scenarios minimizing custom programming; 2) to strengthen the extraction process avoiding non relevant information and correcting OCR errors; 3) to ensure the quality of the extracted information by exploiting the available knowledge to perform a review of the extracted data and curate possible errors; and 4) to allow the integration of additional relevant services (e.g., user validation, image capture tools, help, etc.). To achieve each of these goals, we propose the ICIX architecture, where each objective is achieved as follows:

- (1) the application domain is captured by an ontology (as defined by Gruber [11], a formal and explicit specification of a shared conceptualization), which contains not only knowledge about the domain, but also knowledge about the structure of the different types of documents, as well as references to pertinent extracting mechanisms. This ontology guides the different stages of the extraction process.
- (2) the recall and precision of the extraction is improved by adding a pre-processing step where text curation [7] and automatic text classification are applied.
- (3) the knowledge captured in the ontology is leveraged to detect and solve consistency errors in the extracted data.
- (4) the architecture is designed so that new modules can be assembled allowing additional annex functions.

We have applied our proposal to the legal domain implementing the AIS¹ System, which has been integrated within the commercial Content Relationship Management (CRM) solution of InSynergy Consulting², a well-known IT company, belonging to the International TESSI group³. In this context, we have performed a set of preliminary tests with data extracted by AIS from a real legal document dataset, composed

of notarial deeds of constitution of mortgages (in Spanish), showing the feasibility and the benefits of our approach.

The rest of the paper is structured as follows. Section 2 briefly describes the ICIX architecture. Section 3 provides design details of the implementation of the processing documents task in a specific context. Section 4 presents the results of the empirical study conducted to assess our proposal over that context. Section 5 gives an overview of the state of the art concerning our approach. And finally, Section 6 offers some concluding remarks and directions for future work.

2 ARCHITECTURE OVERVIEW

Figure 1 provides an overview of the ICIX architecture. First, we will focus on its inputs and outputs, to then describe the proposed information repositories and give a brief explanation of the document processing steps. Finally, we will comment some possible relevant services that can be added to the architecture.

2.1 Inputs and Outputs

The main input of the system are the documents containing the information that has to be extracted. Such documents are provided by users of the system in their daily work, and are assumed to be text-based. However, they may present some difficulties, such as including irrelevant content (titles, page numbers, headings, etc.), being very long, or even containing overlapping elements as they have been digitized (signatures, stamps, etc.). The output is the set of specific data extracted from each of these documents, which is stored along them as metadata.

2.2 Information Repositories

As we can see in Figure 1, we have two main knowledge and data repositories in ICIX: the Knowledge Base, and the Document Database.

Knowledge Base: It is the main element that backbones all the rest of the ICIX architecture, and stores the knowledge which guides the extraction process. Such an ontology must model and capture the following elements:

- *Document taxonomy and structure:* The different possible types of documents are classified hierarchically in a taxonomy. Each document class contains information about the sections that shape their kind of documents. Moreover, each of these sections includes further information about which properties and entities have to be extracted from them. This information comprises different aspects apart from just inclusion (e.g., constraints that the elements extracted from a section must hold), making it possible to check the validity of the extraction results.
- *Entities and extraction methods:* The ontology also stores which entities have to be obtained, how they should be processed, and how they relate to other entities. Besides, the key attributes (i.e., the set of attributes which defines an entity uniquely) for each

¹AIS stands, in Spanish, for *Análisis e Interpretación Semántica* (Analysis and Semantic Interpretation).

²<http://www.isyc.com>

³<https://www.tessi.fr>

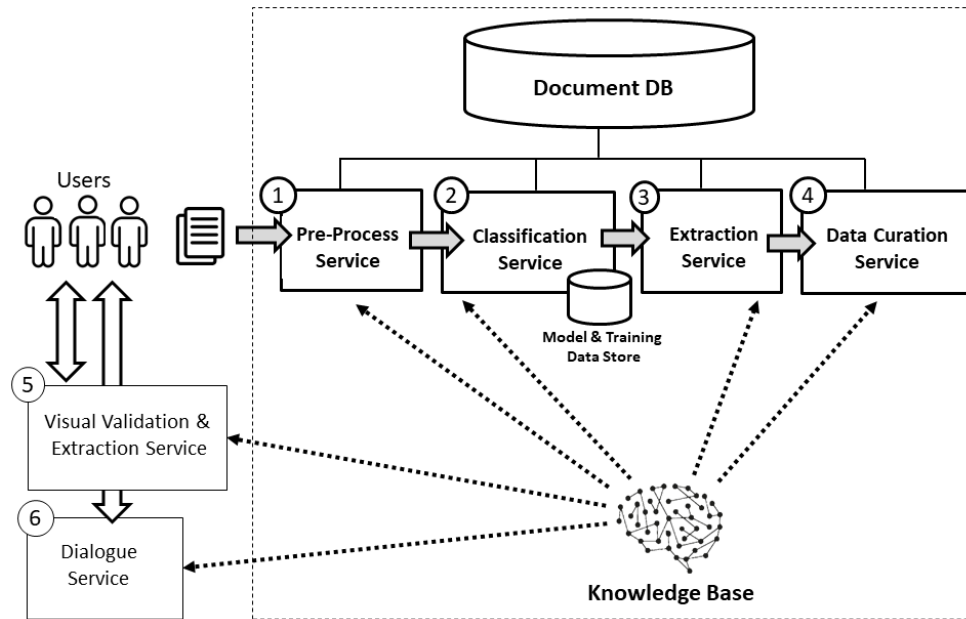


Figure 1: Overview of the ICIX architecture. The dotted box limits the information extraction system. Each box is one of the internal services of the proposed architecture, all of them connected to the knowledge base. Services 1-4 are those that perform the extraction process itself and 5 and 6 are examples of complementary services that can be coupled to the system. The grey arrows represent the document data flows, and the white arrows represent the users' interaction with useful additional services. To apply ICIX in different contexts, it is only necessary to modify the Knowledge Base and the Database.

of the entities are marked as such (e.g., the name, the surname, and the national identity document of a entity *Person*). This decoupled knowledge makes it possible to reuse and adapt easily the extraction operations developed for each entity among different types of documents.

- *Entity instances*: Finally, the ontology also stores information about previously extracted and curated data as instances of the different entities defined in the model. This information is the extensional knowledge that our proposed approach uses.
- *Configuration data*: The ontology also contains the information necessary to parametrize other aspects of the different systems, thus easing the general configuration of the system and its adaptation to different contexts.

The Knowledge Base is connected to all the processing modules, and as we will see below, it is the knowledge that provides them with the necessary information to carry out their work.

Documents Database: It is the information repository where the documents are stored. For each document, it stores two versions of its text: the original, and the cleaned one. Besides, each document is augmented with different attributes (which can be document annotations or the results of an extraction): typology, properties, temporal data, extracted data, etc.

2.3 Document Processing

In Figure 1, we can see the main services that handle the documents sent for processing and subsequent storage (elements 1-4):

- (1) *Pre-process Service*: Even when the OCR is perfect, spelling problems or noisy words can be found. Moreover, the scanned documents can have overlapping elements like stamps and signatures, which further hinders its automated treatment. For solving these problems we propose including a text cleaning and correcting service for automatically recovering data from large documents (text correction is a well-know and broadly applied task in NLP [20]). The configuration parameters of all these tools will be mapped into terms in the ontology. The classification service will use the text obtained in this pre-process step as input.
- (2) *Classification Service*: This service automatically categorizes the content entered by users, using text classification techniques over a set of possible categories which are defined in the ontology. To train the classification model, we build on annotated documents which are obtained by introducing some users in the loop. To avoid classification drifting, we should retrain and redeploy the model as soon as we detect deviations. In this module, we must store the train data and the different trained models.

- (3) **Extraction Service:** This is the core service of ICIX. The previous cleaning and correction steps, as well as knowing the typology of the document at hand, boosts strongly its performance in specific information extraction tasks. Once the service knows the type of document, it queries the ontology to obtain which elements it must look for in the text, and which extraction methods must be used to obtain their related information. The element-value pairs obtained in this process are the basis on which the Data Curation Service works.
- (4) **Data Curation Service:** Once the extraction process is completed, our proposal leverages the knowledge stored in the domain ontology to: 1) perform a review of the extracted data, curating possible errors, and 2) improve substantially the quality of the results by correcting and enriching them exploiting the available knowledge.

At the end of the last stage, all the information about the document (i.e., the text obtained in the pre-processing, the results of the classifier, and the extracted information) are stored in the document database along with it.

2.4 Other Relevant Services

For completeness sake, we show how we can extend our architecture to exploit the extracted data building relevant services on top. In particular, in Figure 1 we can see two services (modules 5 and 6) which provide specific capabilities that complement the process in order to, on the one hand, validate and obtain information using image capture devices, and on the other hand, help and guide the users:

- **Visual Validation & Extraction Service:** This service deals with the treatment of specific documents used to identify/validate the users, such as identity cards, driving licenses, etc. This service has been provided with the infrastructure and integration necessary to implement biometric recognition, combining streaming video capture with Artificial Intelligence (AI) techniques to identify both people and documents.
- **Dialogue Service:** As documentation processing systems can reach a certain complexity, some mechanism of assistance and help to the user is required. ICIX handles this issue by taking advantage of the information available in its knowledge base and allowing to implement conversational agents on top of it. Our proposal includes the inclusion of such an agent which can communicate with the user in a multimodal way. The responses might be in text or video, or directly lead the user to a specific page on the Web. The knowledge supporting this bot is also represented in the ontology.

3 DOCUMENT PROCESSING: LEGAL DOMAIN

The proposed architecture must be grounded to a particular domain and system implementation. To validate our proposal, we have selected the legal domain. Legal documents, such as notarial and judicial acts, or registration documents, are often extensive and the relevant data to be identified for

most of the document management tasks are often few and very specific. Given the importance of having precise results, the identification of these data is still handmade in many organizations. Thus, it is a good application scenario of the ICIX architecture, and it will help us to show the details of the design of the document processing system.

The AIS System: The architecture presented in the previous section has lead the implementation of AIS, the information extraction system integrated in *OnCustomer*⁴, a commercial Content Relationship Management (CRM) system developed by the company *InSynergy Consulting*⁵. The input of the system consists of a set of legal documents in PDF format with unstructured or semi-structured information, and their type. All the documents have been previously scanned and processed by ABBYY⁶, an Optical Character Recognition (OCR) tool. The output is a set of XML files containing structured data extracted from each input text, and which will be stored in a PostgreSQL database. We have used Open Refine⁷ for standardizing, validating, and enriching the extracted data. Regarding the attached systems, the validation service has been implemented with the tools provided by Veridas⁸, and dialogue service has been implemented as a videobot implemented using Dialogflow⁹. All these elements are connected with the ontology, which provides both configuration information and specific data. The following sections provide the most important details, which extend and integrate the proposals presented in [4, 5, 17], works we refer the interested reader to for further details.

3.1 Legal Domain: Document Ontology

Figure 2 shows an excerpt of the ontology that defines, for example, a notarial document:

- Black boxes represent high level definitions that are used regardless of the document typology: document, sections, and entities.
- Gray boxes represent data that are shared among notarial documents regardless of their particular purpose (i.e., the introductory section and the appearances).
- White boxes represent distinctive information of this typology (e.g., the purchased property and its surroundings).
- Dotted boxes represents the extraction methods, and the standardization and enriching operations, which are modeled in the ontology as annotations of the corresponding classes.

Regarding the relations defined in the ontology, straight arrows represent ‘is-a’ relationships, while dashed arrows represent the ‘contains’ relation. This model makes it possible to define a modular knowledge-guided architecture, so new

⁴<http://www.isyc.com/es/soluciones/oncustomer.html>

⁵<http://www.isyc.com>

⁶<https://www.abbyy.com/>

⁷<http://openrefine.org/>

⁸<https://www.das-nano.com/veridas/>

⁹<https://dialogflow.com/>

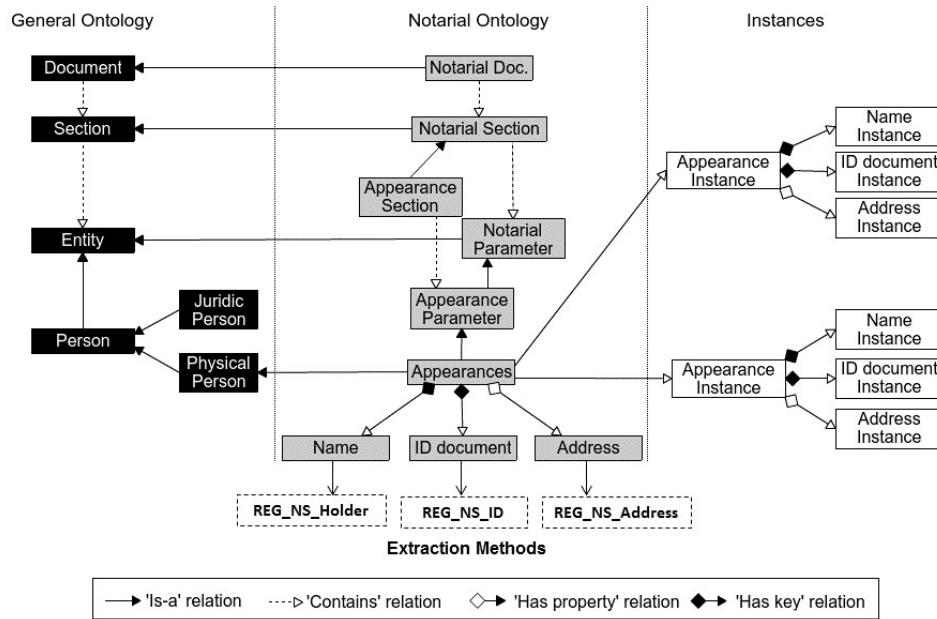


Figure 2: Excerpt of the AIS Ontology showing information related to a notarial deed and references to extraction methods.

document types can be added dynamically to perform the data extraction.

Apart from storing the knowledge about the structure of the documents in the domain, this ontology is extended and populated with the information stored in the repositories of real environments where AIS is currently being used (as well as with previously extracted facts). This is done by means of a periodic process using R2RML mapping¹⁰, which basically maps each of the entities and the properties in the domain ontology to a set of SQL queries which accesses the appropriate data, and allows to format the data according to our ontology, updating the knowledge leveraged by our system.

We have designed and implemented this ontology as an ontology network [27]. We chose this methodology because most documents are classified following a hierarchy with different features. This model allows us to define a modular knowledge-guided architecture, so new document types can be added dynamically. The ontology in our prototype has 25 classes and 33 properties, and information about 2 chunking procedures and 17 extraction methods. The extraction methods used in the experiments are mainly based on symbolic pattern rules due to the good performance empirically obtained with this methodology over this type of documents.

3.2 AIS Pre-Process Service

As we have presented in Section 2.3, due to several factors, we need to pre-process the documents to clean the text in order to improve the extraction performance (noisy text data

and misspelled words hinder the extraction tasks). Thus, in AIS, we have implemented a text correction service which comprises three steps to improve the text quality before the extraction process:

- (1) *Text Cleaning Filter*: The first step of our text curation process consists of removing non relevant information from the input to get the text as clean as possible for the correction step. Non relevant elements are usually page numbers, stamps, noisy characters, etc., which difficult the extraction process, and they are removed by applying regular expressions on the text, provided by the ontology.
- (2) *Generic Text Correction*: The second step is in charge of correcting the misspelled words which appear in the text. For this purpose, we advocate to apply a general purpose spell checker. We have used two open source spell checkers: Aspell¹¹, and JOrtho¹². They have different features and performances, so we have combined them to get better data quality.
- (3) *Specific Text Correction*: Finally, we use an N-gram based spell checker, also generic, but specifically modified for enhancing its performance for the domain of the documents to be processed. We have used Hunspell¹³, a powerful spell checker and morphological analyzer which offers a good multi-language support (e.g., English, French, Spanish, etc.), and we have developed our own N-gram libraries. For each detected error for a given word (*wi*), 1) we get the spell checker suggestions

¹¹<http://aspell.net/>

¹²<http://jortho.sourceforge.net/>

¹³<http://hunspell.github.io/>

¹⁰R2RML: RDB to RDF Mapping Language, <https://www.w3.org/TR/r2rml/>

(si), and, 2) we assign a score based on an adaptation of the Needleman-Wunsch [12] distance and computed with the actual word, and we reinforce them with the probabilities of the N-grams suggestions as follows:

$$wordScore(wi, si) = gSP(wi, si) * NG(wi)$$

where gSP is the score based on the metric distance, and NG is the probability of wi being the next word in the domain where it appears. Both values are normalized in the 0–1 range. Note that our system only accepts words that are suggested by the spell checker, and gets their probability from the N-gram suggestion list. Besides, $wordScore$ is never 0 as $gSP(wi)$ always returns a value greater than 0, because we add perplexity to our N-gram model using the add-one Laplace Smoothing method [15].

The benefits of using this combined approach are two-fold: on the one hand, the general spell checker allows us to leverage all the general purpose techniques that are usually used to perform the corrections; on the other hand, the use of an N-gram-based model allows us to adapt them to the particular domain we are tackling exploiting text regularities detected in successfully processed domain documents. In this case, the word sequences which frequently appear in a particular document typology makes our system to be able to perform a highly adapted word-level correction for that kind of documents. For example, the sequence "notary of the Illustrious College of" have a high probability of being followed by "Madrid" in Spanish notarial documents.

3.3 AIS Classification Service

In our implementation of the text classification service, to model the document text, we have adopted a continuous distributed vector representation for text fragments of variable length, from a sentence to a large document. These vectors, called *Paragraph Vectors*, are obtained through an unsupervised learning algorithm that learns sequence representations adopting the task of predicting words inside or within neighbouring sequences [18]. Using this technique, each document is mapped into a single vector. Besides, as a by-product, a distributed vector representation of the vocabulary in the domain is obtained, adapted to the particularities of the language use in the corpus of documents.

After learning the domain vocabulary and document vector representations/models, we classify each document by means k-nearest neighbors algorithm (k-NN), where the closest documents vote for the actual class of a given one (in this case, we use the cosine distance between document representations, and a k value of just one). Note that once we have enough documents to obtain an initial document representation space, we do not need to retrain it from scratch as this approach allows us to obtain representations of new documents without having to do so. The annotations required to train the document classifier are obtained by asking some particular users to help annotating some seed documents.

3.4 AIS Extraction Service

The data extraction process is guided by the information represented in the ontology and it follows two steps: 1) *Sectioning*, to divide the text into different sections that contain valuable information according to the stored knowledge in the ontology, and 2) *Extraction*, to extract entities from each section also according to the stored knowledge. Sections and entities are both obtained by applying extraction operations over the input text.

As shown in Figure 2, the ontology allows us to specify the extraction methods to be applied depending on the domain and the document typology. Innerly, the operations are defined as tuples $R = \langle T, C, P \rangle$, where T is the type of the operation, C is the context of the operation, i.e., a set of variables v_1, v_2, \dots, v_N that the previous operations have initialized to define the current state of the data extraction task, and P is a list of parameters p_1, p_2, \dots, p_M of the rule, with $M > 0$. This allows to express small IE programs which can be easily reused. These operations slightly follow the line of the Data Extraction Language (DEL¹⁴), but in our case they are oriented to natural language. In more detail, the operations are the following:

- **Segmentation:** Let $text_i$ be the input text, this operation extracts the set $\{seg_j\}$, i.e., the set of segments which conforms the input text. Such segments are defined by two regular expressions, $p_1 = regExp_{start}$ and $p_2 = regExp_{end}$, which mark the start and end of a particular segment. If one of $regExp_{start}$ or $regExp_{end}$ is empty, the segment $\{sect_j\}$ is defined by $[regExp_j, regExp_{j1}]$ or $[regExp_{j-1}, regExp_j]$. This operation also provides different policies which allow to deal with isolated matches, e.g., if the matching sequence $\{start_1, end_1, end_2\}$ is found, the service can choose between forming a segment $start_1, end_1$ or to extend it to $start_1, end_2$.
- **Selection:** Let $text_i$ be the input text, this operation extracts the set $\{text_{ij}\}$, i.e., the set of all subtexts of $text_i$ that match the list of regular expressions $\{regExp_{begin}, regExp_{inside1}, \dots, regExp_{insideN}\}$ defining an IOB format¹⁵.
- **Replacement:** Let $text_i$ be the input text, $regExp_R$ a regular expression, and str a string literal. This operation returns $text'_i$ as the output text that has all the $regExp_R$ matches on $text_i$ replaced by the str literal.
- **Variable manipulation:** Let $text_i$ be the input text, C_i the context of the extraction process before operation i , $regExp_C$ a regular expression, and n a natural number. This operation extracts the set $\{text_{ij}\}$ of all subtexts of $text_i$ that match the regular expression, and binds the result of $size(\{text_{ij}\}) > n$ to the next context C_{i1} .

¹⁴<https://www.w3.org/TR/data-extraction>

¹⁵Inside Outside Beginning (IOB) format is used for tagging tokens in a chunking task in computational linguistics. I indicates that a tag is inside a chunk, O indicates that a token belongs to no chunk, and B indicates that a tag is the beginning of a chunk.

- **Flow control:** Let $text_i$ be the input text, C_i the context of the extraction process before operation i , $\{L_j\}$ a list of sets of operations, and $\{Cond_j\}$ a list of both regular expressions $regExp_j$ or variables v_{ij} defined inside C_i . This operation evaluates $\{Cond_j\}$ sequentially until either a match of $regExp_j$ inside $text_i$ or the value of v_{ij} returns true. Then, it applies the list of operations L_j to the input text.

In addition, this service is able to manage regular expressions extended with NLP analysis which are used to enhance the operations. These extended regular expressions are defined in a similar way as in NLTK¹⁶, i.e., as a list of regular subexpressions $regExp_{EXT} = \{subRegExp_i\}$, being $subRegExp_i$ either a classical regular expression, or a custom regular expression which handles morphological analysis. We extend the capabilities of this process by integrating a Named Entity Recognizer (NER) [21] to identify mentions of entities in the text. Regarding the NER tool, we use Freeling¹⁷, a well known and widely used analysis tool suite that supports several analysis services in both Spanish and English, as well other languages which could be incorporated in our architecture in future developments.

3.5 Data Curation Service

Finally, after the extraction process, AIS curates the extracted data to improve the outcome of the overall information extraction process. This whole process is guided by the knowledge stored in the domain ontology, analyzing the extracted entities, their structure and their attributes. In particular, it performs the following steps:

- (1) *Basic Cleaning:* First, the process cleans empty fields and duplicated information. We dedicate the first step to these two types of errors due to their frequency and their particularly nocive effects in subsequent steps.
- (2) *Data Refinement:* This step is focused on refining those entity attributes whose content is corrupted, e.g., the first name of a person contains the full name of one or more different persons. For each attribute of each entity, this step searches instances stored into the ontology, then, if it detects that an attribute is not valid, it modifies the attribute with the appropriate value stored in the ontology. If it does not find an equivalent value, the process does not change it.
- (3) *Validation and Enrichment:* In this step, the data is standardized, validated, and enriched exploiting different sources of information. Currently, these sources mainly include: 1) previously extracted and verified data included as instances in the ontology, and 2) available external information services (e.g., *Google Maps API*¹⁸). We apply this step to standardize and validate the value of the attributes of the entities with the goal of solving later duplicities. For example, the type of

a street can be extracted as 'St.' or 'Street', but they are representing the same concept; or the service could extract from a certain postal address only the street and the city, but the zip code could be obtained from Google Maps.

- (4) *Entity Division:* Once the data is validated and enriched, this step: 1) detects attributes which belong to different entities but are assigned to a single extracted entity, and 2) separates such attributes assigning them to the appropriate ones. The detection is focused on the set of attributes that uniquely defines an entity (i.e., their key attributes), e.g., the data extracted for a person could include two driving license numbers. In order to detect potentially unified entities, the process searches entities in the ontology using combined key attributes. In case of success, it creates a new entity with such data, and deletes those attributes from the rest of entities. If no entity is found or the attributes of another entity remain in the result, the process creates a number of entities equals to the maximum number of occurrences of a key attribute, and each attribute is assigned to an entity. For this purpose, our approach uses an *Entity Aligner* module which includes different functions and methods to measure the similarity between two entities [2, 10, 13].
- (5) *Final Review:* We include this last step to remove those attributes or information which remain at the end of the process and which have not been assigned to any entity or validated, or do not cover the minimum cardinality specified in the ontology.

Our approach uses intensively the information stored into the domain ontology to support the data curation process in different steps, relying on its already validated information and trusted information services to improve the quality of the extracted data. Note how this approach could be easily adapted to other domains whenever there is available curated information to be leveraged.

4 RESULTS

To evaluate the whole pipeline, we carried out a batch of experiments focused on testing the overall performance of our IE approach. For this purpose, we used a dataset of 250 documents within the legal domain from which we extracted the data by hand to obtain the baseline. To measure the results, we used the well-known classic measures in IE to test the accuracy of a system: *Recall* (R), *Precision* (P), and *F-Measure*. In Table 1, we show the results obtained in terms of *Macro* and *Micro* values:

- *Micro:* The micro values show the accuracy taking into account all the possible data to be extracted, making no distinction among the different entities which the data belongs to. As we can see in Table 1, we obtained a value of 0.82 in F-Measure.
- *Macro:* The macro values are calculated by grouping the accuracy values firstly per entity, and then averaged. It allows to give a more detailed view about all the

¹⁶A toolkit for building Python programs to work with natural language. <http://www.nltk.org>

¹⁷<http://nlp.lsi.upc.edu/freeling/>

¹⁸<https://developers.google.com/maps/documentation/geocoding/start>

Table 1: Performance of the information extraction approach

Type	Similarity	Metric	Value
Micro	-	Precision	0.73
		Recall	0.94
		F-measure	0.82
Macro	75%	Precision	0.82
		Recall	0.84
		F-measure	0.83
Macro	90%	Precision	0.79
		Recall	0.82
		F-measure	0.80
Macro	100%	Precision	0.71
		Recall	0.73
		F-measure	0.72

extracted entities and their values. Moreover, in some scenarios, a little noise in the extracted is acceptable, and we wanted to show how our approach behaves. For this purpose, we used three values depending on a similarity threshold (75%, 90% and 100%). This similarity is computed using the *Needleman-Wunsch* [12] metric which allows us compare the degree of correctness of an extracted entity. In Table 1, we can see the results obtained for each threshold.

These results are obtained using the whole pipeline. In a more detailed analysis of each step, we measured that:

- The document pre-processing allows to improve the hit and miss results in a about a 3%), a slight improvement, but we have to take into account that we work on top of general spell checkers that have been thoroughly trained and tested (it is really robust). Regarding false positives and negatives, the N-gram blocks corrections that are wrongly proposed by spell checkers, achieving a reduction of 13% in false positives, at the cost of an increment of just a 0.5% in false negatives, which can be considered a good result.
- Applying the classification service and the post-processing guided by the ontology, we obtained an improvement in F-measure of about a 12%.

5 RELATED WORK

Information extraction (IE) is the process of scanning text for obtaining specific and relevant information, including extracting entities, relations, and events. Within the broad spectrum of approaches to this generic problem, we could situate our work as a rule-based system guided by an ontology.

The use of ontologies in the field of Information Extraction has increased in the last years. Thanks to their expressiveness, they are successfully used to model human knowledge and to implement intelligent systems. Systems that are based on the use of ontologies for information extraction are called OBIE systems (Ontology Based Information Extraction) [31]. The use of an ontological model as a guideline for the extraction

of information from texts has been successfully applied in many other works [3, 16].

While statistical machine learning is widely used as a black-box technology regarding transparency of decisions, rule-based approaches follow a mostly declarative approach leading to explainable and expressive models. Other advantages of such rule languages include readability, and maintainability and foremost the possibility of directly transferring domain knowledge into rules [30]. The potential of directly including the knowledge of a domain expert into the IE process rather than choosing the most promising training data, hyper-parameters or weights, is a major advantage compared to other approaches to IE.

To the best of our knowledge, there do not exist other works focused on creating a generic architecture/methodology dedicated to the specific extraction of information on documents. Besides, it is hard to locate information about extraction systems equipped with robust extraction methods in the presence of OCR problems due to overlapping marks. Focusing on particular domains, in the recent years, we can find some works in the legal [6, 8, 9, 23], in the industry [19, 25, 26], and medical domains [22, 29], in which the use of ontologies and rules can be deemed similar to ours. The main difference between these aforementioned works and ours is threefold: 1) we aim at a flexible architecture adaptable to any domain, 2) our proposed ontological model captures both the nature of the entities and the formal structure of documents in order to boost the extraction process, and 3) ICIX performs an integral improvement of the extraction results by providing a complete end-to-end pipeline: previous cleaning and correction processes, use of automatic classifiers enabling the suitable extraction tools, the post-verification process, as well as the presence of satellite validation and help tools.

6 CONCLUSIONS

In this paper, we have presented a generic architecture to carry out an automatic data extraction from documents. Our extraction process is guided by the modeled knowledge about the structure and content of the documents. This knowledge is captured in an ontology, which also incorporates information about the extraction methods to be applied in each section of the document. The service-oriented architecture approach we have adopted makes our architecture flexible enough to incorporate different techniques to perform data extraction via invoked methods. The ontology has been upgraded by storing additional domain knowledge in order to detect and solve consistency errors in the extracted data. Besides, we have endowed the architecture with a pre-process service with the aim of correcting OCR error and minimizing the presence of annoying overlapping elements in the document, an automatic classifier which optimizes the choice of extraction methods to use.

The knowledge-based nature of our approach, as it is completely guided by the domain ontology, enhances its flexibility and adaptability, and ensures its portability to other use cases. In particular, its application to other business

contexts would only require to adapt the data and knowledge expressed by the ontology; the rest of the system would remain unchanged.

To show its flexibility and feasibility, we have detailed the AIS system, the implementation of our architecture for the legal domain. The preliminary experiments we have carried out suggest that exploiting knowledge to guide the extraction process improves the quality of the results obtained, obtaining good results regarding precision and efficiency. Despite the fact that experimental dataset is composed of Spanish legal documents, our approximation is generic enough to be applied to documents in other languages. The good results achieved so far lead to optimism regarding the interest of this architecture.

As future work we want to apply our approach to other domains apart from legal one, extending the experiments to other fields. Regarding the architecture, we want to explore the incorporation of external web services to the platform, that allow expanding the functionalities of the approach.

ACKNOWLEDGMENTS

This research work has been supported by the OTRI project ICIX7 2019/0628 at the University of Zaragoza, and the project TIN2016-78011-C4-3-R (AEI/FEDER, UE). We want to thank Maria G. Buey, Javier Plano, and Ruben Jarne for their collaboration in the project.

REFERENCES

- [1] Martin Atzmueller, Peter Kluegl, and Frank Puppe. 2008. Rule-Based Information Extraction for Structured Data Acquisition using TextMarker. In *Proc. of Intl. Conf. of Learning, Knowledge, and Adaptability (LWA 2008)*. 1–7.
- [2] Danushka Bollegala, Yutaka Matsuo, and Mitsuru Ishizuka. 2007. Measuring semantic similarity between words using web search engines. *WWW* 7 (2007), 757–766.
- [3] Javier Rincón Borobia, Carlos Bobed, Angel Luis Garrido, and Eduardo Mena. 2014. SIWAM: Using Social Data to Semantically Assess the Difficulties in Mountain Activities. In *10th International Conference on Web Information Systems and Technologies (WEBIST'14)*. 41–48.
- [4] María G. Buey, Angel Luis Garrido, Carlos Bobed, and Sergio Ilari. 2016. The AIS Project: Boosting Information Extraction from Legal Documents by using Ontologies. In *Proc. of Intl. Conf. on Agents and Artificial Intelligence (ICAART 2016)*. INSTICC, SciTePress, 438–445.
- [5] María G Buey, Cristian Roman, Angel Luis Garrido, Carlos Bobed, and Eduardo Mena. 2019. Automatic Legal Document Analysis: Improving the Results of Information Extraction Processes Using an Ontology. In *Intelligent Methods and Big Data in Industrial Applications*. Springer, 333–351.
- [6] Tin Tin Cheng, Jeffrey Leonard Cua, Mark Davies Tan, Kenneth Gerard Yao, and Rachel Edita Roxas. 2009. Information extraction from legal documents. In *Proc. of Intl. Symposium on Natural Language Processing (SNLP 2009)*. 157–162.
- [7] Edward Curry, Andre Freitas, and Sean O’Riáin. 2010. The role of community-driven data curation for enterprises. In *Linking enterprise data*. 25–47.
- [8] Denis Andrei de Araujo, Sandro José Rigo, and Jorge Luis Victória Barbosa. 2017. Ontology-based information extraction for juridical events with case studies in Brazilian legal realm. *Artificial Intelligence and Law* 25, 4 (2017), 379–396.
- [9] Denis A de Araujo, Sandro J Rigo, Carolina Muller, and Rove Chishman. 2013. Automatic information extraction from texts with inference and linguistic knowledge acquisition rules. In *Proc. of Intl. Conf. on Web Intelligence (WI 2013) and Intelligent Agent Technologies (IAT 2013)*, Vol. 3. IEEE, 151–154.
- [10] Jérôme Euzenat and Petko Valtchev. 2004. Similarity-based ontology alignment in OWL-lite. In *Proc. of Intl. European Conf. on Artificial Intelligence (ECAI 2004)*. IOS press, 323–327.
- [11] Thomas R. Gruber. 1995. Toward Principles for the Design of Ontologies Used for Knowledge Sharing. *International Journal of Human-Computer Studies* 43, 5-6 (1995), 907–928.
- [12] Yaser Jararweh, Mahmoud Al-Ayyoub, Maged Fakirah, Luay Alawneh, and Brij B Gupta. 2019. Improving the performance of the needleman-wunsch algorithm using parallelization and vectorization techniques. *Multimedia Tools and Applications* 78, 4 (2019), 3961–3977.
- [13] Yong Jiang, Xinmin Wang, and Hai-Tao Zheng. 2014. A semantic similarity measure based on information distance for ontology alignment. *Information Sciences* 278 (2014), 76–87.
- [14] Armand Joulin, Édouard Grave, Piotr Bojanowski, and Tomáš Mikolov. 2017. Bag of Tricks for Efficient Text Classification. In *Proc. of Intl. Conf. of the European Chapter of the Association for Computational Linguistics (ACL 2017): Vol. 2*. 427–431.
- [15] Daniel Jurasky and James H Martin. 2000. Speech and Language Processing: An introduction to natural language Processing. *Computational Linguistics and Speech Recognition* (2000).
- [16] Agnieszka Konys. 2018. Towards knowledge handling in ontology-based information extraction systems. *Procedia computer science* 126 (2018), 2208–2218.
- [17] Víctor Labrador, Alvaro Peiró, Angel Luis Garrido, and Eduardo Mena. 2020. LEDAC: Optimizing the Performance of the Automatic Classification of Legal Documents through the Use of Word Embeddings. In *Proc. of Intl. Conf. on Enterprise Information Systems (ICEIS 2020)*. 181–188.
- [18] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31st International conference on machine learning*. 1188–1196.
- [19] Kaijian Liu and Nora El-Gohary. 2017. Ontology-based semi-supervised conditional random fields for automated information extraction from bridge inspection reports. *Automation in construction* 81 (2017), 313–327.
- [20] Christopher D Manning and Hinrich Schütze. 1999. *Foundations of statistical natural language processing*. Vol. 999. MIT Press.
- [21] David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Linguisticae Investigationes* 30, 1 (2007), 3–26.
- [22] Nassim Abdeldjalil Otmani, Malik Si-Mohammed, Catherine Comparot, and Pierre-Jean Charrel. 2019. Ontology-based approach to enhance medical web information extraction. *International Journal of Web Information Systems* (2019).
- [23] Prakash Poudyal and Paulo Quaresma. 2012. An hybrid approach for legal information extraction. In *JURIX*, Vol. 2012. 115–118.
- [24] Erhard Rahm and Hong Hai Do. 2000. Data cleaning: Problems and current approaches. *IEEE Data Engineering Bulletin* 23, 4 (2000), 3–13.
- [25] K Rajbabu, Harshavardhan Srinivas, and S Sudha. 2018. Industrial information extraction through multi-phase classification using ontology for unstructured documents. *Computers in Industry* 100 (2018), 137–147.
- [26] Syed Tahseen Raza Rizvi, Dominique Mercier, Stefan Agne, Stefan Erkel, Andreas Dengel, and Sheraz Ahmed. 2018. Ontology-based Information Extraction from Technical Documents. In *Proc. of Intl. Conf. on Agents and Artificial Intelligence (ICAART 2018)*. 493–500.
- [27] Mari Carmen Suárez-Figueroa. 2010. *NeOn Methodology for building ontology networks: specification, scheduling and reuse*. Ph.D. Dissertation. Universidad Politécnica de Madrid.
- [28] Kees van Noordwijk. 2017. Integrated Legal Information Retrieval; new developments and educational challenges. *European Journal of Law and Technology* 8, 1 (2017), 1–18.
- [29] Natalia Viani, Cristiana Larizza, Valentina Tibollo, Carlo Napolitano, Silvia G Priori, Riccardo Bellazzi, and Lucia Sacchi. 2018. Information extraction from Italian medical reports: An ontology-driven approach. *International journal of medical informatics* 111 (2018), 140–148.
- [30] Bernhard Walzl, Georg Bonczek, and Florian Matthes. 2018. Rule-based information extraction: Advantages, limitations, and perspectives. *Jusletter IT (02 2018)* (2018).
- [31] Daya C Wimalasuriya and Dejing Dou. 2010. Ontology-based information extraction: An introduction and a survey of current approaches. *Journal of Information Science* 36, 3 (2010), 306–323.

Software Quality Assessment of a Web Application for Biomedical Data Analysis

Kristina Lietz
Bioinformatics Group
Fraunhofer ITEM
Hannover, Germany
kristina.lietz@item.fraunhofer.de

Ingmar Wiese
Bioinformatics Group
Fraunhofer ITEM
Hannover, Germany
ingmar.wiese@item.fraunhofer.de

Lena Wiese*
Bioinformatics Group
Fraunhofer ITEM
Hannover, Germany
lena.wiese@item.fraunhofer.de

ABSTRACT

Data Science as a multidisciplinary discipline has seen a massive transformation in the direction of operationalisation of analysis workflows. Yet it can be observed that such a workflow consists of potentially many diverse components: like modules in different programming languages, database backends, or web frontends. In order to achieve high efficiency and reproducibility of the analysis, a sufficiently high level of software engineering for the different components as well as an overall software architecture that integrates and automates the different components is needed. For the use case of gene expression analysis, from a software quality point of view we analyze a newly developed web application that allows user-friendly access to the underlying workflow.

CCS CONCEPTS

• **Software and its engineering** → **Requirements analysis; Maintaining software;** • **Applied computing** → **Computational transcriptomics; Bioinformatics;**

KEYWORDS

Data Science workflow, Gene expression analysis, Software quality, Web service

ACM Reference Format:

Kristina Lietz, Ingmar Wiese, and Lena Wiese. 2021. Software Quality Assessment of a Web Application for Biomedical Data Analysis. In *25th International Database Engineering & Applications Symposium (IDEAS 2021)*, July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3472163.3472172>

1 INTRODUCTION

Data has evolved to become one of the most important assets for any business or institution in recent years and constitutes one of the key factors driving current innovations. For example, the steep spike in technologies for personalized medicine enabled by wide applications of genome or transcriptome analysis are facilitated

by a constant collection of (big) data. Workflows in a Data Science context refer to the processing of data from its raw form to the finished interpretation and result visualization. Such a workflow includes steps such as cleaning, feature engineering, model generation, and validation. In order to fully operationalize the workflow, this process may also include the deployment of the workflow in an interactive webservice.

As a typical use case for an analysis workflow we address gene expression analysis. Gene expression analyses are important in areas such as drug development and tumor research. The challenge of analyzing genome and transcriptome data volumes has become increasingly important in recent years and cover aspects of exploratory data analysis, statistics and machine learning. Since scientists performing gene expression experiments usually lack statistical and computer science knowledge to handle those amounts of data, usable interfaces are needed. To enable an easy analysis of the data, we designed and implemented a web service that abstracts from the mathematical details and any programming tasks. The web interface was implemented using the R Shiny framework which was chosen because R libraries are widely used for gene expression analysis. Yet so far the R Shiny framework itself has not been thoroughly evaluated regarding its capability for implementing complex data science applications.

As the main goal of this article we provide an evaluation of our web application in terms of software quality. More precisely, our focus lies on comparing the maintainability of modularized versus non-modularized Shiny applications as exemplified by our gene expression web application.

A minor secondary goal of the article is to cover the combination of aspects with respect to workflows and DevOps by briefly describing an implementation of an IT system infrastructure that serves as our basis for a flexible deployment of the data analysis workflows and web applications. We believe that a consideration of DevOps paradigms can simplify development and operations of typical Data Science workflows and enables a high quality in terms of reproducibility, reusability and automation be achieved with low maintenance effort.

Outline. The paper is structured as follows. Section 2 describes the gene expression analysis workflow as a use case. Section 3 introduces the topic of software quality. Section 4 summarizes our requirements analysis and presents related work. Section 5 discusses our approach towards modularity of our application. Section 6 presents general design principles and analyzes their analogies in Shiny. Section 7 gives an in-depth analysis of our application based on quality metrics. Last but not least, Section 8 touches upon

* Also with Institute of Computer Science, Goethe University Frankfurt.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 978-1-4503-8991-4/21/07.

<https://doi.org/10.1145/3472163.3472172>

aspects of a real-world deployment before concluding the article in Section 9.

2 USE CASE: GENE EXPRESSION ANALYSIS

Gene expression is defined as the process of using the sequence of nucleotides of a gene to synthesize a ribonucleic acid (RNA) molecule, which in turn triggers the synthesis of a protein or performs some other biological function in a cell. As a result of this process, the nucleotides' sequence determines the biological information of a gene [6].

2.1 Data Format

We focus on development of a web service to analyze gene expression data generated with Affymetrix microarray chips. Microarrays are a collection of genes or cDNAs arranged on a glass or silicon chip [4]. DNA molecules are located on spots or features on the microarray chips' silicon surface. On each feature are a few million copies of the same section of a DNA molecule. This section can be associated with a specific gene and is in Affymetrix chips composed of 25 nucleotides. cDNA molecules get labeled with a fluorescent dye and form the target molecule of the experiment. Finally, the intensity of the fluorescence of the individual spots is measured using special lasers. The more the oligonucleotides in the spots are hybridized, the stronger the emission of light will be. An image is generated from the intensity of the individual spot's fluorescence and stored for further analysis. The generated image is kept in the data (DAT) file format, which contains the measured pixel intensities as unsigned integers. A so-called CEL file summarizes the DAT file and is used in the further analysis.

Beyond the measurement data, additional metadata annotations have to be stored. The MicroArray Gene Expression Tabular (MAGE-TAB) specification is a standard format for annotating microarray data, which meets MIAME requirements. MIAME is a recommendation of the Microarray Gene Expression Database society. It is a list of information that should be provided at a minimum when microarray is published to allow description and reproduction [5]. MAGE-TAB defines four different file types: Investigation Description Format; Array Design Format; Sample and Data Relationship Format (SDRF); raw and processed data files. For the subsequent analysis, the SDRF file is of particular importance because information about the relationships between the samples, arrays (as CEL files), and data of the experiment is contained.

2.2 Workflow

The data analysis for a gene expression experiment is usually conducted by following a specific workflow. After performing a gene expression experiment, first of all the quality of the resulting data must be verified using different metrics. If the quality is insufficient, some data may need to be removed from the analysis or parts of the experiment may need to be repeated [13]. If the quality of the data is sufficient, the data needs to be preprocessed. This process consists of three steps: Background correction to remove influences on the data that have no biological cause; normalization to make individual samples comparable; and summarization to calculate one value from multiple measured values of the activity of the same

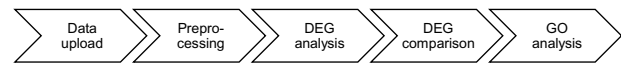


Figure 1: Workflow for analyzing differentially expressed genes (DEGs)

gene [13]. Finally, the actual statistical data analysis is performed to answer the research question [15].

We present now the workflow from the user's point of view by detailing all steps followed by a user in our web application. These steps are initially derived from the MaEndToEnd workflow [17], yet several customizations had to be added as requested by the endusers of our web service. In Figure 1, the basic steps are visualized: Data upload, preprocessing, DEG analysis, DEG comparison and gene ontology (GO) analysis.

- (1) The workflow starts with the upload of the raw input files. For this, the user has to define the source of the files, currently either the local file system, a SQL database connection or the GEO repository [10] are available as options. It is possible to filter the samples from the input data since not all samples may be of interest.
- (2) As the first step of the preprocessing, the uploaded files can be filtered for their so-called gene chip types. An all-in-one pre-processing algorithm can be selected by the user, which is then applied to the data. It performs the steps of background correction, normalization, and summarization. Afterwards, the user can plot the data in various ways for quality control.
- (3) Next follows the actual gene expression analyses. A histogram of the distribution of the p-values is plotted for each analysis, where it is expected that the frequency ranges between very high near 0 and low towards 1. The user verifies this. If the histogram does not meet the expectations, the workflow ends at this point. Possible faults could be incorrect data, incorrect pre-processing or improperly defined analyses. If the histogram meets the expectations, the next step is to set the significance thresholds. These define ranges of statistical values calculated during the analysis in order to identify differentially expressed genes (DEGs).
- (4) In the next step of the workflow, it is possible to compare the performed gene analyses with each other. For the comparison, different plots and Venn diagrams should be created automatically.
- (5) For classifying the identified differentially expressed genes (DEGs) in the biological context, gene ontology (GO) [7] based enrichment analyses are performed in our web service. This offers a uniform vocabulary, which applies to all eukaryotes.

Our web application offers a separate pane for each of the above workflow steps.

3 SOFTWARE QUALITY

As already described, the importance of data analysis applications is increasing. Whether a framework is suitable for the development of

complex applications depends on many factors. We aim to analyse our web service according to standardized software quality metrics. The term software quality is defined by the International Organization for Standardization/International Electrotechnical Commission (ISO/IEC) as the “degree to which a software product satisfies stated and implied needs when used under specified conditions” [16]; see also [12] for a recent survey. The ISO/IEC 25000 norm represents an international standard entitled “Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE)” including a section on System and software quality models. A distinction is made between the following quality models: *Quality in Use* and *Product Quality*. We will focus here on Product Quality that is defined as “characteristics [...] that relate to static properties of software and dynamic properties of the computer system” [16].

We now introduce a selection of these characteristics that we later consider in the assessment of our software.

- (1) Functional suitability consists of the sub characteristics functional completeness, appropriateness, and correctness. It addresses the effectiveness of the software product and is assessed for our software by a comprehensive requirements analysis and subsequent evaluation in user tests.
- (2) Usability includes appropriateness recognizability, learnability, operability, user error protection, UI aesthetics, and accessibility. Thereof, appropriateness recognizability is defined as “the degree to which users can recognize whether a product or system is appropriate for their needs”. Usability of our software is also evaluated in user tests and feedback interviews.
- (3) Maintainability is defined as “the degree of effectiveness and efficiency with which a product or system can be modified by the intended maintainers”. It includes the sub characteristics modularity, reusability, analyzability, modifiability, and testability. The sub characterizations are explained further because they are considered in more detail to evaluate our Shiny framework later on with quantitative metrics.
 - (a) Modularity describes the separation of the software into components, where changes in a component should affect dependent components as little as possible.
 - (b) Reusability means the use of assets in several software systems or with building new assets. Assets are defined as “work products such as requirements documents, source code modules, measurement definitions, etc.” [16].
 - (c) The sub characteristic analyzability describes the understanding of the interrelations in the software product, for example, the impact a change would have.
 - (d) Modifiability describes the extent to which software can be changed without degrading quality or introducing errors. According to ISO/IEC, the modifiability can be impacted by modularity and analyzability.
 - (e) Finally, testability describes the possibilities of setting up test criteria and verifying their compliance.
- (4) Compatibility (within the characteristic co-existence) describes the impact on other software running on the same platform as well as (within the characteristic interoperability) the exchange of information with other software. These

aspects are considered in our software since input and output file formats are supposed to be interchangeable with other software products by using standardized data formats, allowing access to public repositories (like Gene Expression Omnibus) as well as offering a versatile database system access.

We mention in passing that the Product Quality model [16] also covers other characteristics which are however out of the scope of this paper. The characteristics of performance efficiency describes the properties time behavior, resource utilization and capacity and depends on the used platform or environment. Reliability describes the maturity, availability, fault tolerance and recoverability of the software. Security includes the confidentiality, integrity, non-repudiation, accountability and authenticity of the software, which will not be considered in this article since the application is only accessible to researchers inside the institute. Finally, portability includes the adaptability, installability and replaceability of the product; while not quantitatively analyzing portability, we address this issue in Section 8.

4 REQUIREMENTS ANALYSIS AND COMPARISON TO RELATED WORK

For the implementation of the web service, the software requirements have to be specified first. During the complete development process, new requirements were identified, resulting from literature search and interviews with future users of the web service. As already mentioned, various requirements were derived from the MaEndToEnd workflow. Requirements were identified and documented during the entire development process.

A distinction between functional and non-functional requirements is made. Examples of non-functional requirements are the system’s availability or safety aspects. The requirements are graded into high, medium and low concerning their relevance for implementation [24]. We identified 30 functional requirements and 4 non-functional requirements (which cannot be fully reproduced here in detail due to space restrictions); out of the functional requirements 13 and out of the non-functional requirements 2 were graded as high.

Because of the relevance of gene expression analyses, many tools for those are already available. For justifying the development of a new tool, the following paragraphs will give a short presentation of a selection of existing tools and show the differences between the defined requirements and the functionalities of these tools. The Transcriptome Analysis Console (TAC) software [1] is presented because Fraunhofer ITEM researchers currently use it for gene expression analyses. The other presented tools were selected based on a literature search; two of them were chosen because their functionalities and the defined requirements intersected as closely as possible: GEO2R [20] and Network-Analyst [25]. In particular, only tools that are open source and offer a graphical UI were selected.

The TAC software [1] is provided by Affymetrix Inc., the manufacturing company of the microarray chips. The gene expression analyses are mostly performed with the R packages provided by Bioconductor [14]. According to our requirements analysis we identified the following shortcomings of the software: Uploading or creating SDRF files is not possible, but definable attributes provide

the functionality. Furthermore, it is possible to compare several analyses, but not to the required extent. An installation of the TAC software on the client is mandatory. The significance values cannot be set individually for the single DEG analyses. Also, performed analysis steps can only be traced by manually repeating the steps.

GEO2R offers an option for gene expression analysis available on the website of the GEO database [10]. According to our requirements analysis we identified the following shortcomings of the software: Local files and SDRF files cannot be uploaded. Furthermore, only one biological group per sample can be specified, which limits the possible analyses. GEO2R does not offer the ability to generate PCA plots and heat maps or filter the samples after quality control. Multiple DEG analyses can only be defined indirectly because all defined biological groups are automatically compared. For these, only similar significance thresholds can be defined. A comparison of several analyses is only offered for analyses made between samples of the same series and only via one Venn diagram.

NetworkAnalyst [25] is a free web-based tool for gene expression analysis that offers DEG analyses and network analyses. According to our requirements analysis we identified the following shortcomings of the software: Files in CEL format, as well as SDRF files, cannot be uploaded. However, the functionality of defining biological groups can be achieved by specifying the factors elsewhere. For preprocessing, the tool only applies the filtering of genes and the normalization step. Background correction, as well as summarization, does not seem to be performed. Moreover, the filtering of samples after quality control is not possible. Multiple DEG analyses can be defined, but not as precisely as required and individual significance thresholds can only be defined for the adjusted p-value. Several analyses can be compared, but not to the requested level.

Overall, no tool completely meets the requirements graded “high”. Especially the comparison of analyses is only possible to a limited extent. The development of an own application for gene expression analysis within our institute offers further advantages. Data that has not yet been published can be analyzed without any concerns about data protection. In contrast, the security of third-party tools, especially if they are publicly available on the internet, has to be verified first. Moreover, if new requirements arise, the software can be flexibly extended.

5 SHINY MODULARIZATION

The programming language R provides an environment for the analysis and graphical visualization of data [22]. R was chosen for our implementation because the open-source software project Bioconductor provides many R packages for the analysis of gene expression data. Although these could also be used in other programming languages, the application’s complexity can be reduced by limiting it to one programming language. This has a positive impact on the maintainability of the software. With R Shiny, a framework for the development of web-based applications directly in R is provided. An application is built from two main components: A UI object and a corresponding server function, which accesses the UI elements via their defined IDs. Also, the server function defines the logic for the functionality of the web app, for example, processing the data or plot variables. The elements defined in the UI object are either input or output elements. Values of the input

elements are processed in the server function code, while the output elements visualize the results of those calculations. Moreover, Shiny uses a reactive programming model. Reactivity in Shiny means that when the input values are changed, the code sections that access the changed input elements are automatically re-executed. Thus, the corresponding output elements are also automatically updated.

As mentioned in Section 4, for the implementation of our web service, we first specified software requirements by close communication with the end users. During the complete development process, new requirements were identified, described, and evaluated. The process was thus iterative. A first prototype of our framework hence resulted in a *monolithic* source code.

Defining modules should improve the handling of complex Shiny applications. For this purpose, functions are used, which are the fundamental unit for abstraction in R. Modules consist of functions that generate UI elements and functions used in the module’s server function. There is a global namespace within a Shiny application, so each ID of input or output elements must be unique within the app. By using functions for generating UI elements, the uniqueness of the IDs must be ensured. Shiny modules solve this problem by creating an abstraction level beyond functions.

Shiny modules have three basic characteristics:

- (1) They cannot be executed alone but are always part of a larger application or module. The nesting of modules is therefore possible.
- (2) They can represent input, output or both.
- (3) They are reusable: both in several applications and several times in the same application.

For creating a module, one function is needed that defines UI elements and one function that uses these elements and contains the server logic. The UI function expects an ID as a parameter, which defines the namespace of the module. The caller of the function defines this. The namespace is applied to all IDs of the input and output elements for ensuring the correct mapping of the elements to the namespace. This ID as namespace must also be passed to the server function of the module. In this, the `moduleServer` function is invoked, where the actual server logic is defined. The objects input and output are aware of the namespace, so the elements with ID wrapped in the defined namespace can be referenced using the standard way `input$elementID`. UI elements outside the namespace cannot be addressed at all. In the regular components of the Shiny application, the UI function of the module is called in the UI object and the server function of the module in the regular server function. It is essential to pass the same ID to the corresponding functions.

In order to improve maintainability, we refactored the monolithic version of our app into a modularized one. Figure 2 illustrates an overview of the defined main modules and their sub modules of the web application. One module was defined for each main step of the workflow, resulting in the following main modules: `dataUpload`, `preprocessing`, `degAnalysis`, `degComparison`, and `goAnalysis`. Each of these main modules are realized within a separate tab/panel in the web application. Each of the main modules accesses at least one sub-module with more specific responsibilities.

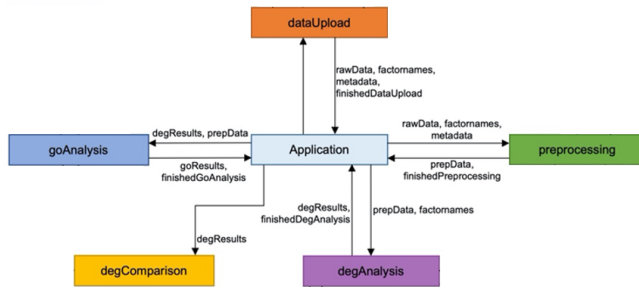


Figure 2: Modularized application

	maintainability	sub characteristics			
		modularity	reusability	analysability	modifiability
encapsulation	↑	↑	↑		↑
coupling	↓	↓	↓	↓	↓
cohesion	↑	↑	↑	↑	↑
size	↓			↓	↓
complexity	↓		↓	↓	↓

Table 1: Influence of OO design principles on the sub characteristics of the software quality property maintainability; an arrow pointing up (↑) indicates a positive influence of the design property on the quality (sub) characteristic and an arrow pointing down (↓) indicates a negative influence.

6 SOFTWARE DESIGN PRINCIPLES

This article focuses on the evaluation of the maintainability of Shiny applications. The focus is on the maintainability of the Shiny components that build and control the user interface (UI). As already specified, maintainability consists of the sub characteristics modularity, reusability, analysability, modifiability, and testability. The latter is not considered in this evaluation of Shiny, as this is a very extensive topic and would exceed the scope of this paper. In the following, a relationship between these sub-characteristics and design principles from the object-oriented (OO) programming paradigm is first established. In this section, the term application refers to a Shiny web app and the term module refers to a Shiny module.

6.1 Influence of OO design principles on maintainability

When designing OO software products, several design principles are defined, supporting the enhancement of the software quality. In the following, a selection of the design principles defined by Schatten et al. [21] is presented and assumptions about their influence on the sub characteristics of maintainability are made. Only the principles relevant to the metrics calculation in Section 7 are considered. The assumptions about the impact on maintainability are based on the results of a literature review and the comparison of the definitions of the design principles to those of the maintainability characteristics.

Table 1 indicates with upward and downward arrows an overview of our assumptions made for the influence of the design principles

on the software quality; that is, ideally higher encapsulation, looser coupling, higher cohesion, and less extensive and less complex software. We discuss our assumptions on software design principles in detail:

Encapsulation. With encapsulation, details of the implementation should be hidden from the outside, realized in the OO paradigm by private fields and methods. Communication between components should occur only via defined interfaces, which are public fields and methods. For example, this enables to change the encapsulated functionality of a component, without affecting the of calling components as long as the interface remains unchanged. The definition’s similarity of the sub characteristic modularity of maintainability is noticeable [16]. Therefore, the assumption is made here that higher encapsulation also increases modularity, reusability, modifiability, and maintainability overall.

Coupling. Coupling describes the dependency of two components, whereby in general, the lowest possible coupling should be achieved. Thus, the effects of changes of a component are reduced in the dependent components [21]. Also, tight coupling complicates the analysability of the software. Therefore, with increasing coupling, the maintainability of the software decreases.

Cohesion. Cohesion describes the degree to which the elements of a component are interrelated. As cohesion increases, the coupling between the individual components usually decreases [21]. Therefore, rising cohesion generally supports the maintainability of the product.

Size and Complexity. In the following, also metrics are considered, which are assigned to the design properties size and complexity. The assumption is made that with increasing size, especially the software’s analysability and modifiability deteriorates while increasing complexity also impairs reusability.

6.2 Transfer of OO concepts to Shiny

Our aim is to analyze our Shiny app according to widely used OO software quality metrics [18]. Because in general the application modules in Shiny do not fully follow the the OO paradigm, some assumptions have to be made to calculate the metrics. For this purpose, a definition of the fields and methods of different encapsulation levels is made for the Shiny framework. Only the elements of a class that are relevant in the metric selection of Terragni et al. [23] are considered. Hence, for transferring the class elements of the OO paradigm to the Shiny framework in a meaningful way, the *class elements*, their functionality and possible encapsulation levels are presented next.

An object or instance has a defined state (defined by attributes or fields) and a defined behavior (defined by methods). In general, the behavior depends on the state, which means that the methods access the attributes. A class defines for a collection of objects the structure, the behavior and the relationship to other classes. Static elements of a class are the same for all instances of that class. Static methods can only be applied to the whole class and not to a single instance. Usually, they are used to assign a new value to class attributes without the influence of an instance or whenever an operation applies to all or several instances. The following

OO paradigm	Shiny framework
Fields	
Static fields	Fields with equal value for all instances; lockBinding in the environment
Private fields	Reactive Values (including Input object) defined inside application or module
Protected fields	-
Public fields	Fields declared in the R script outside of functions; not locked in the environment
Methods	
Static methods	-
Private methods	Functions defined inside server function
	Reactive expressions
	Observers (including render functions)
Protected methods	-
Public methods	Functions defined in the R script outside of the server function
	If application viewed: only server function
	If module viewed: UI and server function

Table 2: Mapping of OO class components to Shiny applications and modules

three levels of encapsulation are differentiated: private elements are accessible only within the class itself; protected elements are accessible in the class itself, in its subclasses and, depending on the programming language, in classes within the same package; public elements are accessible by all classes. The following statements can be made by transferring these definitions to the Shiny framework. Instances of a Shiny application or module are distinguished from each other by using different namespaces. An instance of an application or module also consists of a state and behavior. The state is primarily defined by input elements used by the defined behavior to calculate the output elements. The definition of the application or module determines the state and the behavior for a collection of instances. *Static fields* should have the same value for all instances of an application or module. *Static functions* should be applicable to all instances. The latter is not implementable for Shiny applications and modules because the principle of object management is not realized by default. However, the object management could be achieved using static attributes, but this was not implemented in the exemplary software, so that static methods in the Shiny framework will not be considered in further discussion. The distinction of instances based on the namespace allows two stages of encapsulation to be defined: Private elements are only accessible within their own namespace, and public elements are also accessible outside the namespace. The encapsulation level *protected* of the OO paradigm has no correspondence in Shiny since inheritance is not implementable in this framework.

We now discuss how the individual elements from the OO paradigm are mapped to elements of applications and modules of the Shiny framework base on these assumptions; an overview of this mapping is provided in Table 2.

It is assumed that *private fields* correspond to reactive values defined within the application or module in the Shiny framework. Reactive values are used in the defined behavior of the application

or module to generate the output. Moreover, they are not accessible outside the namespace of the application or module. Fields that are defined outside of a function in the script of an application or module and are not locked in the global environment are seen as *public fields*. These elements can also be addressed outside the namespace itself. They might also be assumed as static, but any instance can change the value of the field. This may not affect all other instances of this application or module that already exist, especially in other sessions. Therefore, only fields with the same value for all instances and are locked in the environment via the lockBinding function are assumed to be *static fields*. A field has the same value for all instances if the value does not depend on any reactive values or method parameters of the functions. Locking the field prevents the assignment of a new value. Thus, the definition of static variables in the Shiny framework is stricter than in the OO paradigm, but otherwise, the equality of the field's value for all instances cannot be ensured. *Private functions* in the Shiny framework correspond to functions defined within the server function of an application or module, reactive expressions and observers, including render functions. These functions define the behavior of the application or the module by calculating the output in dependency of the state of an instance. Functions defined in the server function are not accessible outside of the namespace of the application or module. Observer and render functions can only be defined inside other functions, so they are automatically not callable from the outside. Also, they are not explicitly invoked in the program code anyways but are executed based on reactivity. Finally, *public functions* are functions defined in the R script outside of any other functions. For Shiny applications, this does not include the server function since this cannot be called outside the application in any meaningful way, except to create a new instance. For Shiny modules, the UI and server functions are also included as they provide the module's interface and are thus called by other modules or applications.

7 SOFTWARE QUALITY ANALYSIS

This section examines how the software quality characteristic maintainability, which was defined in Section 3, is affected by the modularization of Shiny applications by comparing modularized and non-modularized version of our application. With the help of the calculation of software quality metrics, the effect of the modularization of Shiny Apps on maintainability is considered quantitatively. These metrics are mostly derived from the OO paradigm, so they must be partially adapted to the Shiny paradigms. As such, a formal evaluation of the use of the Shiny framework for the development of larger, more complexly structured applications is made. We rely here only on very specific metrics for a static code analysis – which is an established method [9, 11, 23] – due to the fact that we compare two versions of our application with a fixed feature scope. An advanced assessment could also use a cost model based on change rate of the code; in [2] the authors argue that this cost model enables the prediction of future development costs.

7.1 Metrics calculation

Based on the correspondences between Shiny and the OO paradigm just discussed, the effect of modularizing Shiny applications

(see Section 5) is examined. For this purpose, the presented application for gene expression analysis was implemented from an earlier development state both with modularization and without (monolithic). Any comments or other code documentation were removed in the source codes of both applications to increase their comparability; this is based on the assumption that differences in comments do not influence maintainability. Since both applications represent the same functionality, the actual calculation logic was abstracted into functions called by both applications. In the following, these methods are referred to as functionality methods or functions. Thus, the source code to be compared contains as far as possible only instructions concerning the UI as well as its control. For the quantitative comparison, metrics were calculated, which were mostly originally developed to evaluate classes of OO programming languages [23]. First, it will be discussed which of the metrics will not be calculated. The number of bytecode instructions (NBI) was used in [23] because applications developed in Java were considered. Compiling R program code into bytecode is possible, but an interpreter is used by default. Therefore, this metric is not calculated. We compare both our applications (monolithic versus modularized) without considering the code documentation, thus it would not be useful to calculate the lines of comment (LOCCOM) metric. The number of static methods (NSTAM) cannot be calculated since it is not possible or reasonable to declare a method of an application or a module as static in Shiny. Moreover, the concept of inheritance is not implementable in Shiny applications or modules, so any metrics that are based on inheritance cannot be calculated. This includes the depth of inheritance tree (DIT), the number of children (NOC), the measure of functional abstraction (MFA), the inheritance coupling (IC) and the coupling between methods (CBM). There is no equivalent in the Shiny framework to the encapsulation level protected used in the OO paradigm, as explained earlier. Therefore, the number of protected methods (NPROM) cannot be calculated.

Next, the metrics that were calculated for the sample applications are presented. The details of how they were calculated will also be discussed. Table 3 provides an overview of the calculated metrics and a brief description.

The lines of codes (LOC) metric counts the number of non-blank lines of the application or the module. The IntelliJ Plugin MetricsReloaded was used for the calculation of the LOC. This metric is used to evaluate the design property *size* and, therefore, has a low value. This applies to all metrics of this design property. The number of public methods (NPM), the number of fields (NOF) and the number of static fields (NSTAF) were calculated by counting the number of corresponding elements in the application or the module. Terragni et al. do not define which encapsulation should be considered for the calculation of the NOF. It is assumed that only public fields are counted since a separate metric exists for private fields. The number of method calls (NMC) is the sum of the number of method calls internal (NMCI) and the number of method calls external (NMCE). Thereby, the NMCI was calculated by counting all invocations of a function defined in the application or module itself. The NMCE is the number of all function invocations defined in another module or an external R package. Also, the calling of functionality methods is included in this metric.

Design Property Name		Description
Size	Lines of Code (LOC)	Number of non-blank lines
	Number of Public Methods (NPM)	Number of public functions in an application or module
	Number of Fields (NOF)	Number of public fields in an application or module
	Number of Static Fields (NSTAF)	Number of static fields in an application or module
	Number of Method Calls (NMC)	Number of function invocations
	Number of Method Calls Internal (NMCI)	Number of function invocations of function defined in the application or module
	Number of Method Calls External (NMCE)	Number of function invocations of function defined in other modules or packages
Complexity	Weighted Methods per Class (WMC)	Sum of the Cyclomatic Complexity of all function in the application or module
	Average Method Complexity (AMC)	Average of the Cyclomatic Complexity of all function in the application or module
	Response For a Class (RFC)	Number of functions that response to a message from the application or module itself
Coupling	Coupling Between Object classes (CBO)	Number of other modules or packages that an application or module is coupled to
	Afferent Coupling (Ca)	Measure of how many other applications or modules use the specific application or module
	Efferent Coupling (Ce)	Measure of how many other modules or packages are used by the specific application or module
Cohesion	Lack of Cohesion in Methods (LCOM)	Difference between the number of function pairs without and with common non-static fields
	Lack of Cohesion Of Methods (LCOM3)	Revised version of LCOM
	Cohesion Among Methods in class (CAM)	Represents the relatedness among functions of an application or module
Encapsulation	Data Access Metrics (DAM)	Ratio of the number of private fields to the total number of fields
	Number of Private Fields (NPRIF)	Number of private fields of an application or module
	Number of Private Methods (NPRIM)	Number of private functions of an application or module

Table 3: Descriptions of the calculated (class) metrics [23]

Next, metrics for quantitative assessment of the design property *complexity* are presented. For all metrics, a low value corresponds to low complexity. The weighted methods per class (WMC) is calculated by the summation of the Cyclomatic Complexity [19] of all functions defined in the application or module, which counts the linear-independent paths within a program. The R package cyclocomp was used to calculate the Cyclomatic Complexity of a single function. The metric average method complexity (AMC) is the average of the Cyclomatic Complexity of all functions in the application or module. Therefore, the AMC_m of the application or module m is calculated by the formula $AMC_m = WMC_m / NPM_m$. The response for a class (RFC) is the number of functions that respond to a message from the application or module itself. It is calculated by counting the number of distinct functions that are invoked by the application or module, no matter if the function was defined inside the application or module itself or not. Functions called multiple times are counted only once [8].

The metrics for measuring the property *coupling* are presented next. Low values are indicative of loose coupling, while high ones indicate tight coupling. The coupling between object classes (CBO) is calculated by counting the distinct afferent and efferent modules or packages of an application or module [8]. Since the example applications do not have circular dependencies, this is the same as the sum of the afferent coupling (Ca) and the efferent coupling (Ce). Ca is calculated by counting the number of modules or applications that call the target module's functions or application. On the other hand, Ce is calculated by counting the modules or packages from which the target application or module calls functions. For example, if the application A invokes a function of module B, the Ca of B, as well as the Ce if A, increases by one. In the considered example, no packages are included for the calculation of Ca since packages usually do not call any functions of a Shiny application or module. In the calculation of Ce, applications were not regarded because no application or module calls another application.

Now, the metrics for evaluating the design property *cohesion* are discussed. The possible and desired values differ between the metrics. The lack of cohesion in methods (LCOM) corresponds to the difference between the number of function pairs without and with common non-static fields. The LCOM of an application or module m is calculated by $LCOM_m = b - c$, where b equals the number of function pairs that do not reference to similar non-static fields and c equals the number of function pairs that do reference to at least one similar non-static field [8]. Fields that are referenced indirectly by the function of interest via invoked functions are also counted. If the result is negative, LCOM is set to 0. This metric's value should be as low as possible, whereby a value of zero indicates a cohesive application or module. However, the evaluation of a value depends on the total number of defined functions in a component. Therefore, the metric LCOM3 was developed to address this problem of the lack of possibility of comparison. The LCOM3 of an application or module m is calculated by $LCOM3_m = ((x - f \cdot a)) / ((a - f \cdot a))$, where f equals NPM_m , a equals the sum of NOF_m and $NPRIF_m$ and x equals the sum of the number of referenced non-static fields of all functions. If only one function is defined in an application or module or has no non-static fields, LCOM3 cannot be calculated and is set to 0. The value of LCOM3 is always between 0 and 2, where 0 indicates a high cohesion. Values above 1 are critical since this shows the

existence of values that are not accessed by any function within this application or module (Henderson-Sellers, 1996, p. 147). The metric cohesion among methods in class (CAM) of an application or module m is calculated by $CAM_m = p / ((y + f))$, where p equals the sum of the number of different types of method parameters of each function defined in the application or module, f equals the sum of NPM_m and $NPRIF_m$ and y equals the number of distinct types of method parameter of all functions in the application or module. A type of a parameter is thereby the class a parameter should inherit, for example, a list, an integer or a data frame. Also, local variables defined in the surrounded function of a private function are considered to be parameters. The range of CAM is between 0, which indicates low cohesion and 1, indicating high cohesion [3].

Finally, the calculated metrics for the assessment of the design property *encapsulation* are presented. The number of private fields (NPRIF) and the number of private methods (NPRIM) is calculated by counting the corresponding elements in the application or module. The data access metrics (DAM) correspond the ratio of NPRIF to the total number of fields defined in an application or module.

7.2 Results of metrics and evaluation

The explained metrics were calculated for the modularized and non-modularized software. For the non-modularized software, the metrics were calculated collectively for the scripts run.R, ui.R, and server.R. These three scripts of the modularized software were also calculated as a group referred to as application. For modules of the modularized software, the metrics were calculated individually in each case. The sum and average of the values of the individual modules were calculated for each metric to optimize the comparability. For the design properties of coupling and cohesion, the calculation of the sum of the modules is not meaningful because these are a measure of the dependency on a component or within the component itself.

The differences in results between the modularized and non-modularized software are now considered for each design property. Thereby the relationship to the sub-characteristics of the maintainability is again addressed. An overview of the results is shown in Table 4. For the modularized software, in addition to the sum and the average, the results of only the application (app) itself are also shown.

The sum of the metrics results associated with the design property size is mostly larger for the modularized software than the non-modularized software results. Exceptions are the NOF and the NMCI and the NSTAF, for which there is no difference between the considered software. Looking at the average of the individual modules, they are all smaller than the monolithic application. The larger sum of the modularized application can be explained because at least two public methods are defined as almost every module interface. This increases the NPM, the NMCE and therefore also the NMC and the LOC. With the increasing size of software, the analyzability and thus modifiability decreases. The metrics of the design property size indicate worse maintainability of the modularized application as a whole in comparison with the monolithic software. However, the individual components are substantially more compact and individually better maintainable. The same statement can be made for the calculated metrics of complexity. The

Design Property	Metric	Modularized			Not Modularized
		Only App	All Modules (Sum)	All Modules (Avg)	
Size	LOC	68	1067	50.81	865
	NPM	0	54	2.57	0
	NOF	0	0	0	2
	NSTAF	0	1	0.05	1
	NMC	48	934	44.48	858
	NMCI	10	17	0.81	40
	NMCE	48	917	43.67	818
Complexity	WMC	8	210	10	148
	AMC	1.6	1.76	1.76	1.78
	RFC	29	207	23.05	168
Coupling	CBO	9	-	4.33	13
	Ca	0	-	0.95	0
	Ce	9	-	3.38	13
Cohesion	LCOM	0	-	7.9	2593
	LCOM3	0.59	-	0.34	0.96
	CAM	0.6	-	0.44	0.07
Encapsulation	DAM	1	0.99	0.99	0.95
	NPRIF	8	81	3.86	63
	NPRIM	5	65	3.1	83

Table 4: Results of the calculated metrics for the application in the non-modularized and modularized design

sum of the individual modules indicates a higher complexity of the entire modularized software than the non-modularized software. On average, however, the individual modules are again less complex than the non-modularized software. The increased WMC could also be caused by the increased number of functions, since each function has a Cyclomatic Complexity of at least 1. Therefore, the AMC is more meaningful for assessing the complexity, which indicates that the modularized software functions are marginally less complex. When calculating the sum of the RFC, the called functions were only counted once across all modules. Again, this higher value can be explained by the higher number of functions in the modularized software. Also, concerning complexity, the statement can be made that the modularized application has overall worse maintainability than the non-modularized software. Once again, however, the single modules are less complex and better maintainable. As already explained initially, only the average value of the modules is compared with the monolithic software for the properties coupling and cohesion. Thereby all computed values show a looser coupling as well as higher cohesion with the modularized software. The only exception is Ca, but this is since the monolithic software as an application cannot be called from outside. As discussed before, looser coupling increases all sub characteristics of maintainability. The increasing cohesion improves the loose coupling additionally and therefore has a positive effect on the maintainability too. Finally, the metrics used to evaluate the software design property encapsulation are evaluated. Here, it has to be said that encapsulation is unnecessary for monolithic architectures since the application is not called from the outside. Nevertheless, the calculated metrics indicate good encapsulation of the modules

because especially DAM has a very high value. This increases the sub-characteristic modularity, reusability and modifiability. Altogether based on the metrics, the modularized software as an entire system is larger and more complex than the monolithic software. This indicates a worse analysability, modifiability, reusability and thus maintainability in total. However, the individual modules could also be regarded separately from each other, whereby the computed metrics indicate substantially higher maintainability. In addition to the quantitative evaluation, the results are now also discussed via a qualitative approach. Although modularization increases the overall size and complexity of software, it also improves its maintainability. However, this depends on the modularization implementation – a correct amount must be defined to have cohesive modules on the one hand and loosely coupled modules on the other hand. This allows a significant improvement of the reusability, analysability and therefore also modifiability of the software. Also, a well-chosen modularization offers the advantage of allowing the modules to be considered separately from one another. The regarded section of the software during the maintenance becomes much smaller and less complex than with a monolithic application. Also, the effect of changes within a module in other modules can be estimated and intercepted much better. Modules can be called multiple times in one or more applications, minimizing or even preventing redundant program code. Altogether with a qualitative view, the modularization simplifies the maintainability of the software substantially.

8 DEPLOYMENT

In a modern DevOps context pipelines usually concern automation from source code to the deployment of the finished product. Pipelines are omnipresent in today's IT landscape. From data pipelines to integration and testing pipelines up deployment pipelines, the term finds broad acceptance and a wide definition. In our system architecture, the steps considered are building, testing, packaging, and deploying a piece of software. We followed this DevOps paradigm in our software development process as follows.

The web service is executed in a Docker container on a server provided by the Fraunhofer ITEM. This server has 48 cores and 256 GB random access memory (RAM) and thus provides sufficient computing power to analyze the gene expression data. The program code is stored in a GitLab repository. A Dockerfile is used to define the build and deployment of the service and define and install prerequisites and dependencies. In turn, the building and pushing of the image into the Docker registry is controlled via a GitLab CI/CD file. This file enables continuous integration (CI) and continuous deployment (CD), meaning running a defined pipeline as well as deploying the application to production whenever a modification is made. The pipeline consists of a test stage for checking the Dockerfile and a deploy stage for deploying the service to the server. Using Portainer, a dashboard for the management of Docker containers, the container for deployment is created and maintained.

9 CONCLUSION

The Shiny applications found in the literature are often quite small and used as dashboards for data visualization. For handling the challenge of developing large applications, we investigated whether this framework is suitable for the development of complex applications.

Our analysis focuses on the formal evaluation of the maintainability of Shiny applications. We provided a comparison of modularized and non-modularized application version and followed a quantitative approach based on chosen software quality metrics. Quantitatively, modularized applications are larger and more complex than monolithic applications. However, the modules can be considered independently so that the resulting maintainability of modularized applications is rated better overall. Most software quality metrics found in the literature are developed to evaluate object-oriented (OO) principles. Therefore, a subgoal of our article was the investigation of the influence of OO principles on the maintainability of software. To be able to transfer these principles to Shiny, the equivalence of Shiny application elements to elements from the OO paradigm needed for the computation of the metrics was examined. For this, the purpose of the OO elements and components of Shiny with the same or similar purpose were investigated. Based on these assumptions, the software quality metrics were calculated. Our application is located in the biomedical realm; hence the choice for Shiny is rooted in the availability of libraries for the backend pipeline in the R programming language. We believe that our discussion generalizes to other web applications that follow a workflow character and that relies on tabs in the web pages for which Shiny components could be reused. Nevertheless, other domains that rely less on specific R backend libraries and that might have a focus on the microservice paradigm might profit more from alternative web frameworks (for example, based on Javascript).

In future work, we aim to add capabilities to execute workflows on other input file formats. A major research question in this regard is whether the modularization of our software proves to support the modifiability aspect.

ACKNOWLEDGEMENTS

This work was supported by the Fraunhofer Internal Programs under Grant No. Attract 042-601000.

REFERENCES

- [1] Affymetrix, Inc. Transcriptome analysis console (tac) 4.0.2 user guide, 2019.
- [2] T. Bakota, P. Hegedűs, G. Ladányi, P. Körtvélyesi, R. Ferenc, and T. Gyimóthy. A cost model based on software maintainability. In *2012 28th IEEE International Conference on Software Maintenance (ICSM)*, pages 316–325. IEEE, 2012.
- [3] J. Bansiya and C. G. Davis. A hierarchical model for object-oriented design quality assessment. *IEEE Transactions on software engineering*, 28(1):4–17, 2002.
- [4] V. Bolón-Canedo and A. Alonso-Betanzos. *Microarray Bioinformatics*. Springer, 2019.
- [5] A. Brazma, P. Hingamp, J. Quackenbush, G. Sherlock, P. Spellman, C. Stoeckert, J. Aach, W. Ansorge, C. A. Ball, H. C. Causton, et al. Minimum information about a microarray experiment (miame)—toward standards for microarray data. *Nature genetics*, 29(4):365–371, 2001.
- [6] T. A. Brown. *Genomes 4th edition*. Garland science, 2018.
- [7] S. Carbon, E. Douglass, B. M. Good, D. R. Unni, N. L. Harris, C. J. Mungall, S. Basu, R. L. Chisholm, R. J. Dodson, E. Hartline, et al. The gene ontology resource: enriching a gold mine. *Nucleic Acids Research*, 49(D1):D325–D334, 2021.
- [8] S. R. Chidamber and C. F. Kemerer. A metrics suite for object oriented design. *IEEE Transactions on software engineering*, 20(6):476–493, 1994.
- [9] D. Coleman, D. Ash, B. Lowther, and P. Oman. Using metrics to evaluate software system maintainability. *Computer*, 27(8):44–49, 1994.
- [10] R. Edgar, M. Domrachev, and A. E. Lash. Gene expression omnibus: Ncbi gene expression and hybridization array data repository. *Nucleic acids research*, 30(1):207–210, 2002.
- [11] J. E. Gaffney Jr. Metrics in software quality assurance. In *Proceedings of the ACM’81 conference*, pages 126–130, 1981.
- [12] T. Galli, F. Chiclana, and F. Siewe. Software product quality models, developments, trends, and evaluation. *SN Computer Science*, 1(3):1–24, 2020.
- [13] R. Gentleman and W. Huber. Processing affymetrix expression data. In *Bioconductor Case Studies*, pages 25–45. Springer, 2008.
- [14] R. C. Gentleman, V. J. Carey, D. M. Bates, B. Bolstad, M. Dettling, S. Dudoit, B. Ellis, L. Gautier, Y. Ge, J. Gentry, et al. Bioconductor: open software development for computational biology and bioinformatics. *Genome biology*, 5(10):1–16, 2004.
- [15] H. Gohlmann and W. Talloen. *Gene expression studies using Affymetrix microarrays*. CRC Press, 2009.
- [16] International Organization for Standardization. *Systems and Software Engineering: Systems and Software Quality Requirements and Evaluation (SQuaRE): Measurement of System and Software Product Quality*. ISO, 2016.
- [17] B. Klaus and S. Reisenauer. An end to end workflow for differential gene expression using affymetrix microarrays. *F1000Research*, 5, 2016.
- [18] M. Lanza and R. Marinescu. *Object-oriented metrics in practice: using software metrics to characterize, evaluate, and improve the design of object-oriented systems*. Springer Science & Business Media, 2007.
- [19] T. J. McCabe. A complexity measure. *IEEE Transactions on software Engineering*, (4):308–320, 1976.
- [20] National Center for Biotechnology Information. <https://www.ncbi.nlm.nih.gov/geo/info/geo2r.html>.
- [21] A. Schatten, S. Biffl, M. Demolsky, E. Gostischa-Franta, T. Östreicher, and D. Winkler. *Best practice software-engineering: Eine praxiserprobte Zusammenstellung von komponentenorientierten Konzepten, Methoden und Werkzeugen*. Springer-Verlag, 2010.
- [22] P. Teetor. *R cookbook: Proven recipes for data analysis, statistics, and graphics*. O’Reilly Media, Inc., 2011.
- [23] V. Terragni, P. Salza, and M. Pezzè. Measuring software testability modulo test quality. In *Proceedings of the 28th International Conference on Program Comprehension*, pages 241–251, 2020.
- [24] P. Voola and A. V. Babu. Comparison of requirements prioritization techniques employing different scales of measurement. *ACM SIGSOFT Software Engineering Notes*, 38(4):1–10, 2013.
- [25] G. Zhou, O. Soufan, J. Ewald, R. E. Hancock, N. Basu, and J. Xia. Networkanalyst 3.0: a visual analytics platform for comprehensive gene expression profiling and meta-analysis. *Nucleic acids research*, 47(W1):W234–W241, 2019.

A Zone-Based Data Lake Architecture for IoT, Small and Big Data

Vincent-Nam Dang
IRIT, (CNRS/UMR 5505)
CNRS
Toulouse, France
vincent-nam.dang@irit.fr

Imen Megdiche
IRIT, (CNRS/UMR 5505)
Institut National Universitaire Jean-François Champollion
Toulouse, France
imen.megdiche@irit.fr

Yan Zhao
IRIT, (CNRS/UMR 5505)
Université Toulouse 1 Capitole
Toulouse, France
yan.zhao@irit.fr

Franck Ravat
IRIT, (CNRS/UMR 5505)
Université Toulouse 1 Capitole
Toulouse, France
franck.ravat@irit.fr

ABSTRACT

Data lakes are supposed to enable analysts to perform more efficient and efficacious data analysis by crossing multiple existing data sources, processes and analyses. However, it is impossible to achieve that when a data lake does not have a metadata governance system that progressively capitalizes on all the performed analysis experiments. The objective of this paper is to have an easily accessible, reusable data lake that capitalizes on all user experiences. To meet this need, we propose an analysis-oriented metadata model for data lakes. This model includes the descriptive information of datasets and their attributes, as well as all metadata related to the machine learning analyzes performed on these datasets. To illustrate our metadata solution, we implemented a web application of data lake metadata management. This application allows users to find and use existing data, processes and analyses by searching relevant metadata stored in a NoSQL data store within the data lake. To demonstrate how to easily discover metadata with the application, we present two use cases, with real data, including datasets similarity detection and machine learning guidance.

CCS CONCEPTS

• **Information systems** → *Database design and models.*

KEYWORDS

Stream IoT Data, Data Lake, Zone-based, Metadata, Technical Architecture

ACM Reference Format:

Vincent-Nam Dang, Yan Zhao, Imen Megdiche, and Franck Ravat. 2021. A Zone-Based Data Lake Architecture for IoT, Small and Big Data. In *25th International Database Engineering & Applications Symposium (IDEAS 2021)*, July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 9 pages. <https://doi.org/10.1145/3472163.3472185>

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](https://permissions.acm.org).

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Association for Computing Machinery.

ACM ISBN 978-1-4503-8991-4/21/07...\$15.00

<https://doi.org/10.1145/3472163.3472185>

1 INTRODUCTION

IoT data is increasingly integrated into the core of today's society. To analyze a market or a product, decision makers must integrate these different IoT data but also combine them with massive data produced internally as well as externally data such as Open Data. To obtain a complete vision, it is therefore necessary to integrate both voluminous fast data and numerous small data.

Faced with the plurality of data types, the data lake (DL) is one of the most appropriate solutions. DL is a big data analytics solution that allows users to ingest data in its raw form from different sources and prepare it for different types of data analysis [11]. With a DL, all types of data can be stored for further analysis for different users. Even if, in principle, the DL seems to adapt to different needs, it is necessary to define an architecture that takes into account all the usual needs of big data analytics focused on large volume data or data with high velocity while integrating the constraints specific to IoT and small data.

To meet the previously stated objectives and contrary to other proposals [6], we propose a complete solution composed of a functional architecture, a technical architecture of a Multi-Zone Data Lake (MZDL) and experiment assessments. This Data lake must allow (i) ingest of different data sources, (ii) implement of pre-processing upstream of analytics, (iii) provide access and consumption of this pre-processed data and finally (iv) properly govern the data throughout previous steps to avoid inaccessible, invisible, incomprehensible [1] or untrustworthy data. To address the veracity problem, we have added an efficient metadata management system that allows the data characterization and security mechanisms and tools that allows greater confidence in the data. This solution will be described from a logical and physical point of view.

The paper is organized as follows. In section 2, a state of the art on IoT data analysis is presented and it shows the limitations of current work. Our proposal is explained in the following sections. In section 3, we provide a functional architecture based on 4 zones. We focus on a metadata model which is not a simple data catalog but which saves all the information related to ingested data, transformation processing and analyzes. This metadata model will facilitate the task of a data analyst by allowing him to easily search for ingested data or even transformation processing or analyzes. In section 4, we expose the different components of our technical architecture and we justify our choices. A discussion about Hadoop is also presented.

Our proposal is experimented in the specific context (NeOCampus project). In section 5, we explained the ingested source of data, the transformation processes we implemented and the analyzes carried out. Finally, we present the metadata management system dedicated to data analysts.

2 RELATED WORK

According to the recent survey on Big data for IoT [4], we notice that most of work deal with only one domain application for example Healthcare, energy, building automation, smart cities and so on. For each domain, the strategy is to collect data coming from several sensors related to the study domain. This IoT big data collected from sensors are different from other big data as they are heterogeneous, various and grow rapidly. Since the definition of big data includes also the possibility to cross several type of data, the multi-domain sensors seems to be much more difficult to resolve than one-domain sensors.

Our second observation concerns the storage and analytics performed on IoT Data. Commonly the storage of IoT data is done on the cloud [13][4]. Concerning the analytics, we identify the following categories according to [7] : (i) real-time analytics, (ii) offline-analytics also named batch analytics, (iii) memory-level analytics, (vi) BI analytics and (v) Massive analytics. The most popular type of analytics are real-time and batch analytics. The Batch analytics does not require quick response, Hadoop framework is the most known framework used in this area with its storage HDFS systems and the MapReduce parallel computing paradigm. For real-time analytics, we distinguish two categories of real-time : strict and flexible. Strict real time is necessary for critical applications and requires specific data management architectures. Flexible real-time (or near-real time) allows freedom in terms of delays and the maximum delays are greater than strict real-time. Our study concern this type of applications. The IoT data handled in this case are called stream data [5]. They arrive continuously with a high velocity. These data are eventually stored and analyzed in architectures such as Lambda architectures or Kappa architectures.

From our both observations concerning the general work on IoT Big Data, the best solution seems to be to use the data lake paradigm in order to capture multi-domain sensors and multiple type of analytics. DL, previously emerged only as a data repository [3], is today one of the most popular big data analysis solutions. A DL ensures data ingestion, raw and processed data storage, data preparation by different users and data consumption for different types of analyses [11].

In the literature, we identify some papers dealing with the problem of storage and analysis of IoT data under the angle of data lake. These papers can be divided into : (i) zone-less data lake architectures and (ii) zone-based data lake architectures.

In the category 'Zone-less IoT data lake architectures', the authors of [6] present a big data lake architecture for multilevel streaming analytics based on Hadoop. We distinguish relational and stream twitter ingestion flow in a single Hadoop/HDFS storage space. The zone-less architecture lacks clear reuse strategy namely without metadata on the processes.

In the category 'Zone-based IoT data lake architectures', the authors of [9] propose a big data lake platform for smart city application. The platform can be considered as zone-based architecture and strongly relies on Hadoop Ecosystem with data ingestion, data storage, data exploration, analytics and data visualization zones. Two level of data ingestion are performed (manual and automatic). The stream data ingestion is realized in near-real time with Flume framework. Concerning metadata implementation, the authors use simple mechanisms rather related to technical implementation. They use also simple embedding metadata information related to file names and paths and metadata description files. The authors of [10], propose a four zone-based data lake architecture for smart grids analytics on the cloud. They collect smart meter, images and video data in the domain of smart Grids. Their architecture is based on the Lambda architecture with both near-real time and batch analytics. This work presents technical implementations and experimentations but this architecture does not enable reuse in the data lake.

According to the challenges and issues related to the big data IoT [2], the data diversity in data storage and analysis is not completely well addressed in the previous cited works. Moreover, knowledge discovery and computation complexities is a real lack in these works. The lack of metadata is commonly observed in these works yet it is an unavoidable solution. The last challenge concern information security which can be addressed with specific services as well as metadata mechanisms.

Our contribution proposes a complete zone-based data lake architecture addressing the different challenges cited below and applicable to several fields. We use a metadata management system that covers the different steps from ingestion, processing to analysis to better reuse of the data lake data.

3 A ZONE-BASED DATA LAKE : A FUNCTIONAL ARCHITECTURE

In this section we describe our proposed zone-based data lake functional architecture and the metadata model dedicated to the architecture.

3.1 Components

This data lake functional architecture facilitates data analytics by allowing users to find, access, interoperate and reuse existing data, data preparation processes and analyses.

In order to answer to near-real time and batch processing requirements, we propose in Figure 1 our functional Data Lake architecture ingesting both IoT, small and Big Datasets. This architecture is a zone-based architecture proposed by [11]. The different zones are described as follows :

- The **raw ingestion zone** allows users to ingest data and stores these data in their native format. Batch data are ingested and stored in this area in their native format, near-real time data are stored after steam processing if users need to store the data. The scalability of this zone is the main criterion to have in order to be able to handle different volumes of data from different domains.
- The **process zone** allows users to prepare data according to user needs and stores all the intermediate data and processes.

In this area, data are valorized by batch process or stream process. This zone concentrates all the treatments and all the business knowledge applied on the raw data.

- The **Access zone** allows the processed data to be accessed, analyzed, consumed and used in any specific business process. It is necessary to be able to enable the consumption of data for visualization, real-time analytics, advanced machine learning or BI analytics and more generally for decision support systems.
- The **govern zone**, applied on all the other zones, is in charge of insuring data security, data quality, data life-cycle, data access and metadata management. This zone has an important role in all stages of the data sustainability and reuse. It is composed of two sub-zones : (i) metadata storage concerns the raw data, the processes and analyses; We wanted to allow the tracking of the data life cycle, from insertion to consumption, and thus allow a follow-up of the needs and a relevant future reuse ; (ii) security mechanisms allow authentication, authorization and encryption as well as multilevel security policy (for instance external and internal threats). This sub-zone allows also quality of service by monitoring a multilevel resources consumption.

3.2 A Metadata Model for the Data Lake Architecture

It quickly became apparent that there was a potential and a need to use metadata for data durability and, more broadly, for the enhancement of this data. We have therefore decided to create a zone exclusively dedicated to metadata from a functional point of view. This zone aims to be the point of conservation of all the metadata, to provide a true valorization of all the data and to allow the reuse of these data in contexts different from the original context. To achieve this goal, it becomes necessary to follow the life cycle of the data. To do this, it is as necessary to design a data management tool as to manage the data.

3.2.1 Metadata for Data ingestion. Data ingestion, the first phase of data life-cycle in a data lake, concerns ingesting external datasets into a data lake. During this phase, metadata of all the ingested datasets should be collected to ensure the findability, reusability, security of datasets (see white classes in Fig. 2). To do so, different types of metadata should be generated :

- *Metadata of ingestion process* (class *Ingest*) which includes ingestion program source code, execution details, link to users, and the upstream and downstream datasets.
- *Metadata of ingested datasets* that includes basic characteristics (class *DatalakeDataset*), *schematic metadata* (classes *DLStructuredDataset*, *DLStructuredDataset*, *DLUnstructuredDataset* and their components) and *semantic metadata* (classes *Tag*, attribute *DatalakeDataset.description*).
- *Metadata of data veracity* (classes *VeracityIndex*) which is a composite index which combines different measures [12]: objectivity, truthfulness and credibility.
- *Metadata of data security* (classes *SensitivityMark*, *SensitivityLevel*) to protect sensitive or private data.

- *Metadata of dataset relationships* (classes *RelationshipDS*, *AnalysisDSRelationship*) which can be predefined relationships such as the similarity, correlation, containment and logical cluster or user defined relationships such as a common subject.

Data ingestion metadata are only instantiated for batch ingestion. Real-time data are processed directly without passing through the ingestion phase. Note that when users want to back up real-time data, this activity is treated as batch ingestion with additional information *SourceOfStream* to indicate the connected object/machine.

3.2.2 Metadata for Data Processes. To ensure that users can find how data are processed and stored in the DL, our metadata model includes the information of data processes (see green classes in Fig. 2). With the process metadata, users can know (i) *process characteristics* that are the basic information about a process, for instance, who did what and when, (ii) *process definition* (attribute *Process.description*) that explains the context, meaning and the objective of a process, (iii) *technical information* is about the source code and execution information which is useful when users want to know how processes are deployed or when they want to modify or reuse a process, (iv) *process content* concerns coarse-grained transformation operators dedicating to qualify data processing [8].

Data processes metadata are instantiated for both batch and (near) real-time process, the only difference between them is that the source dataset of a batch process is a dataset already ingested in the data lake (class *DatalakeDataset*) while the source dataset of real-time process is an external dataset (class *DatasetSource*).

3.2.3 Metadata for Data Analysis. To facilitate data analytics, we also consider the metadata of analysis (see blue classes in Fig. 2) to help users to understand the nature of datasets, to find existing analyzes and their used models, outputs and evaluation, etc. So that users can choose the most appropriate way to analyze data more efficiently [14]. The data analyzes metadata can be applied to both batch (classes *Analysis* and *Study*) and real-time (class *RealtimeAnalysis* and *Study*) analyzes. For batch analyzes, in our model, we precise the useful metadata of machine learning (classes *Implementation*, *Software*, *Parameter*, *ParameterSetting*, *Landmarker*, *Algorithm*, *OutputModel*, *ModelEvaluation*, *EvaluationMeasure*, *EvaluationSpecification*, *Task*).

4 A ZONE-BASED DATA LAKE : A TECHNICAL ARCHITECTURE

We present our technical data lake architecture in Figure 3. In the following we describe the technical implementation of each zone.

Raw Data Zone. For the raw data storage zone, we decided to implement Openstack Swift (OSS). OSS is an object-oriented storage system which allows any type of data to be stored at low cost with the only consideration being the size of the data. OSS allows us to respond to volume, velocity and variety because it allows us to store data independently of the content and structure of the data, allowing us raw ingestion of any data. OSS is part of the Openstack environment allowing native compatibility with Openstack tools and a large community centered around open-source. Communications are done through a RESTful API, a standard that provides

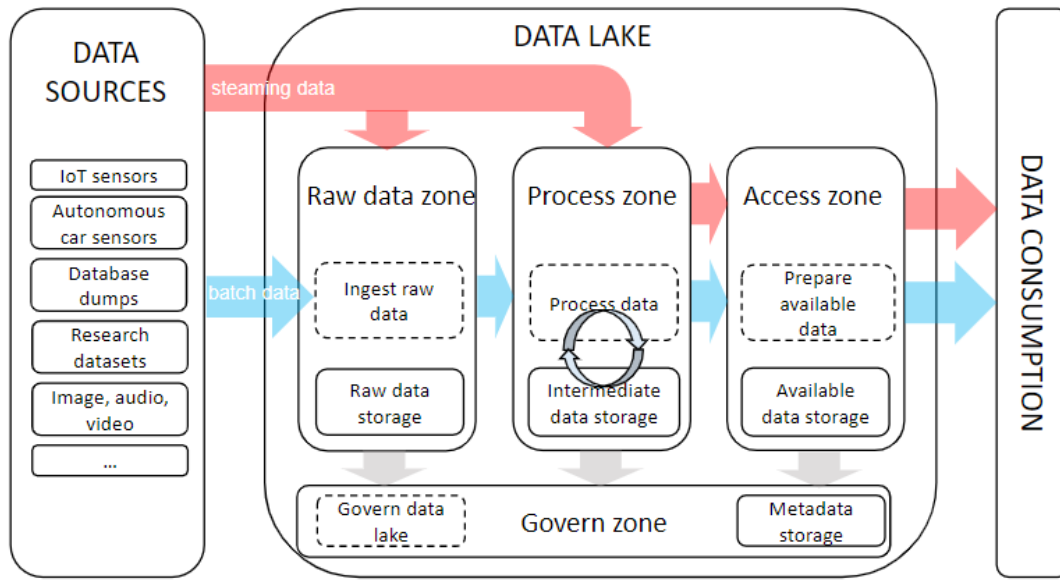


Figure 1: IoT data lake functional architecture

interconnectivity and ease of use. OSS allows linear and elastic scaling of the raw data zone.

Process Zone. The main requirement of the process zone is to allow users to prepare their data and store intermediate results and processes. We chose Apache Airflow because it allows us to rigorously manage the various transformation workflows on the data. This open-source workflow scheduling tool allows to organize and schedule processes in a simple way. This tool allows the implementation of organized processing chain while being very easy to use. It includes a wide variety of tools and programming languages that allow for the definition of transformation functions (for ETL), the use of standard tools from the data analysts' toolbox and the use of large-scale data processing frameworks like Apache Spark. All the processes implemented in this process zone are also stored by OSS in order to easy reuse processes.

Access zone. Once the data are formatted, they can be consumed in the access zone. In order to resolve the interoperability problem with the downstream application of the data lake, we decided to design this zone as a multi-store allowing the implementation of a wide variety of tools through the containerization concept (for instance dockers). This includes all databases, ETL tools, reporting tools and many private or custom tools. The cooperation of this approach with the containerization tools makes it possible to instantiate these tools on the fly, offering great flexibility.

Govern Zone. This zone is composed of two sub-zones : (i) meta-data management sub-zone and (ii) security and monitoring sub-zone.

- The metadata management sub-zone. Metadata, defined with the model described in section 3 are stored in a dedicated data management system. Metadata refers to any metadata over raw or transformed data and processes. The metadata

system solution designed includes a NoSQL graph-oriented database. The advantages of a graph database are multiple: Graph database has a good flexibility, we presented a general metadata model in section 3, it is adapted to different types of datasets for batch and real-time analyses. Moreover, for specific needs, users can always extend to model to fit their requirements. (ii) Graph database provides a powerful manipulation tool which includes not only a standard query language but also integrated machine learning functions. Moreover, when it is easier to query a graph database when the search depth is important. Metadata are updated several steps in the data life cycle : (i) initial metadata are inserted when data ingested and (ii) among all operations done on the data to follow their transformations.

- The security and monitoring sub-zone. From the design stage, we integrate the implementation of security mechanisms, authentication and monitoring of the services allowing users to have a control of the data lake architecture and its resources.
 - For the security, we treat different issues: first we should identify users then we should control their authentication. For identification, we decided to implement Openstack Keystone. This service allows to keep a list of users accounts with token-based authentication (UUID token or Fernet token). Fernet token use AES256 cryptographic protocol and HMAC SHA256 signature verification, creating a secured identification and trustworthy authentication. This is implemented as an input to the architecture in the RESTful API but can be integrated in the pipeline of each service involved via the verification of the validity of the token before any operation. Authentication requires multi-level access control to limit the number of entry points in the architecture and limit the number of tools used. All access are restricted via RestAPI and reverse proxy. One of

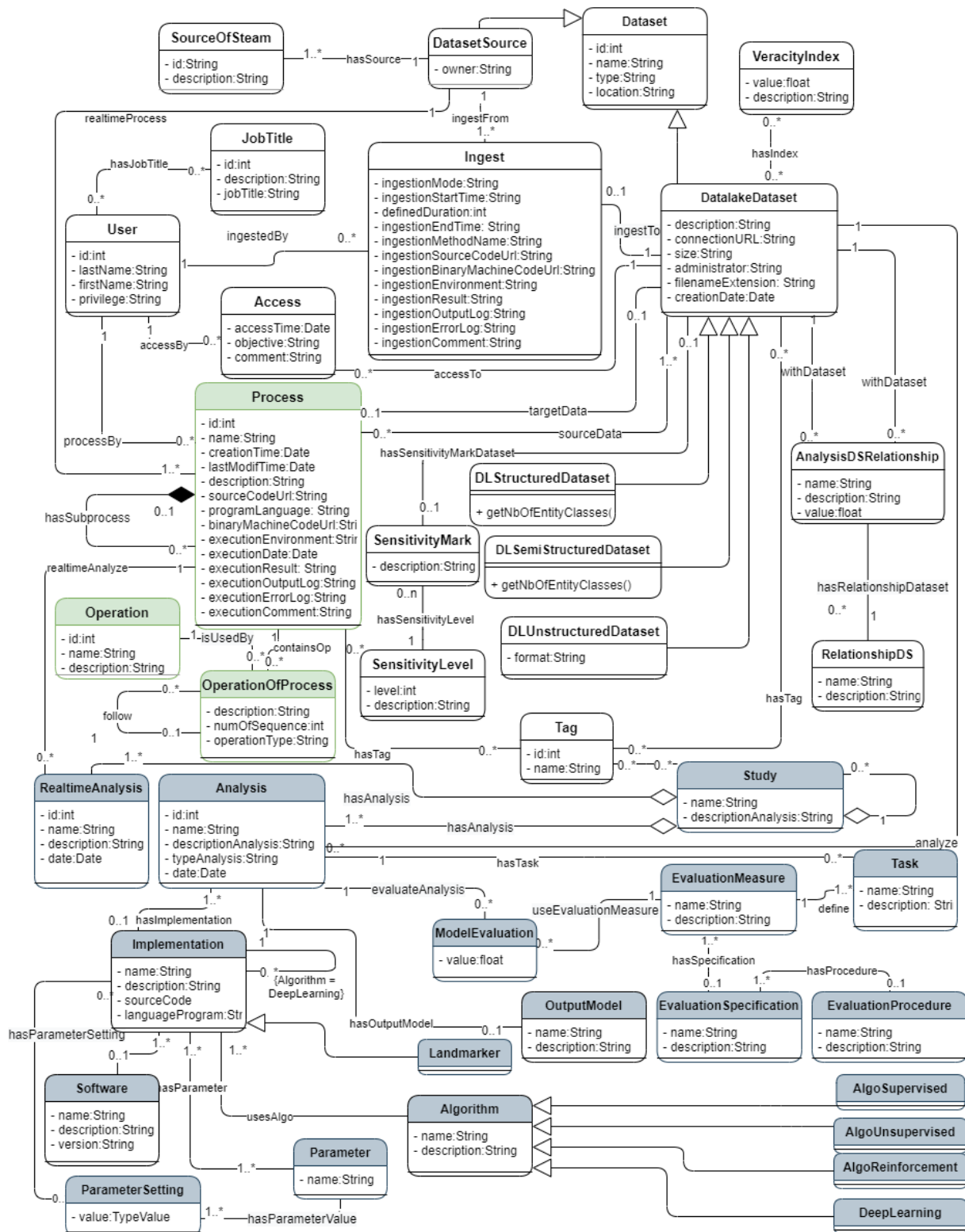


Figure 2: Metadata model

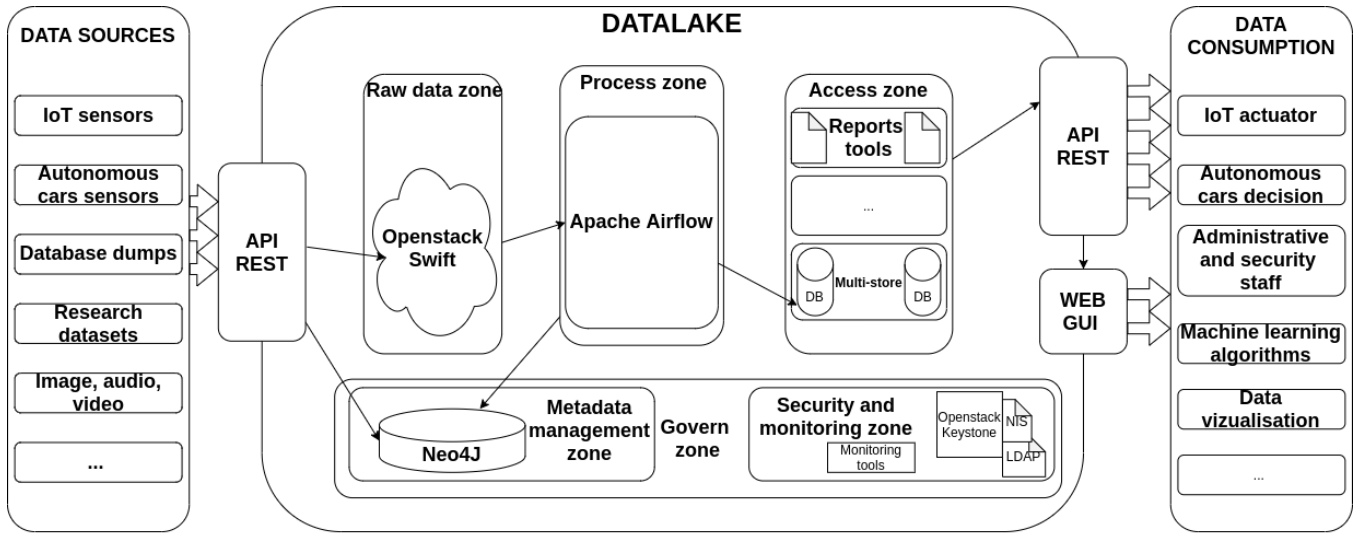


Figure 3: Our big data architecture description

the advantages of Openstack Keystone is the compatibility with very large identification services (NIS, LDAP) and/or authentication services (Kerberos), either natively or by customizing the service. It is thus possible to integrate this authentication system with pre-established systems

- The monitoring concerns is handled at several levels. As it stands, it is necessary to implement three different levels of monitoring. First, the lowest level is a system and network level. The hosting platform defines more precisely the needs and tools to be implemented. Tools such as SNMP, Prometheus and/or Kubernetes monitoring tools are considered to meet security needs. The intermediate level of monitoring is application monitoring. It is necessary to be able to track the resource consumption of each service in the architecture to be able to best scale the platform. Openstack Ceilometer can be a partial solution for this problem. The higher level of monitoring is a monitoring for each user of the resources and services used. This can be used to track meta information on projects and a higher level vision but also to allow a possible billing of services in a private context of service provisioning of this architecture.

Discussion on Hadoop. When it comes to data lake, the most popular solution is the Hadoop ecosystem. This solution is now maintained for many years by the Apache foundation, with a large community, making the solution very appropriate for many use cases. Hadoop has been designed to store and process very large files through a very large parallelization capacity. However, even if it is well adapted to the volumetry, this solution does not seem to be adapted to the other aspects of big data. Indeed, the main problem arises on the variety of data, as we approach it with IoT data, and small data. This problem comes from the design choices of the file system on which the entire ecosystem is based, HDFS. With HDFS, data is divided into data blocks to allow for data replication.

These data blocks are by default 64 or 128 MB in size and are stored on DataNodes¹. Each DataNode keeps track of the data it stores with about 150 bytes of data per data in RAM. The problem arises when the data is smaller than the block size. Thus, the DataNodes' memory fills up before their storage spaces. In most cases, a sensor reading is about 100 bytes. It is difficult to implement a message aggregation policy, which would circumvent this problem, that is efficient, lightweight and does not involve an increase in the complexity of operations performed on this data. Our context requires the management of a wide variety of data, from high volume batch and high velocity stream data to IoT and small data. Data security management and simplicity of deployment of the architecture are important requirements. Thus, we preferred the solution described to the Hadoop ecosystem.

5 EXPERIMENTAL VALIDATION

To validate our solution, we have implemented it for a real project - NeOCampus. In this section, firstly, we introduce the context and details of the project; secondly, we present some results of our implementation.

5.1 Context of the Experimentation

In order to validate our proposal, the data lake architecture² was implemented as a proof of concept (POC) for the NeOCampus project. A specific branch has been created on the repository to track experiment source code used.³ NeOCampus⁴ is a multidisciplinary project gathering many French laboratories with several skills, from computer science to social science. The NeOCampus project is centered around ambient systems and a testing ground to realize an innovative university campus. The ambient systems use many

¹<https://hadoop.apache.org/docs/r3.3.0/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html>

²https://gitlab.irit.fr/datalake/docker_datalake

³https://gitlab.irit.fr/datalake/docker_datalake/-/tree/use_case_paper_IDEAS

⁴<https://www.irit.fr/neocampus/fr/>

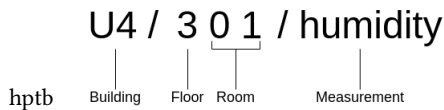


Figure 4: An example of Mqtt topic

sensors leading to a sensor network. The management of this real time IoT data is an important issue for the NeOCampus project. As the project includes several researchers, the management concerns also batch data coming from research projects. Our architecture allows the NeOCampus project to satisfy several objectives such as enabling the reuse of sensors data for research purposes, facilitating brainstorming between several user profiles, encouraging the reuse of processes developed in different projects through the metadata system, cross reference data from several sensors with batch data in a simplified way, create rich dashboards and so on. This experiment has been done on OSIRIM platform with 4 virtual machine from a VMWare virtualisation server. A total of 20 vCPU (1 thread per core, 2.6 GHz each) and 32 Gigabytes of RAM have been allocated.

5.2 Sources of Data

For this POC, we considered two categories of data:

- **Streaming Data** concern the sensor network of the NeOCampus project. In particular we have two equipped rooms in the Paul Sabatier University. The deployed sensors are : (i) luminosity sensor (in lux), (ii) hygrometry sensor (in %r.h), (iii) temperature sensor (degrees are in celsius) , (iv) CO2 sensor (in ppm) (v) energy sensor (complex measurements with several datas). The IoT data arrive at regular intervals, on the order of minutes which lead to a continuous flow of 45 messages per minute corresponding to 105,000 messages in 2 days or 2,000,000 messages per month. Sensor data are collected from the MQTT broker of the NeOCampus IoT network as soon as they arrive. A process subscribes to all topics on the broker to get all sensors reading create. In Figure 4), we have an example of the MQTT topic identification for an humidity sensor.
- **Batch Data** concern research datasets such as the open data of "Météo-France" which contains data collected by about fifty meteorological stations on the French territory. These data are updated every 3 hours and they are available and can be downloaded in CSV files for a defined period (3 hours or over a month). Each CSV file contains 41 columns such as temperature, hygrometry, atmospheric pressure, wind speed, cloud types and their altitude or the type of barometric trend. The data are temporal series indexed by times and station number.

5.3 Data processing

Regarding data processing, we implemented different processing pipelines to process each type of data through Airflow (see Figure. 5). The objective of the processes presented in Figure. 5 is to format the data in order to insert them in a time-series oriented database, InfluxDB.

The choice of pipeline is made by the help of the names of data containers. As the matter of fact, different pipelines share similar operations except for the data parse method. For real-time data, data formatting is applied on each received message sent by sensors. For batch data, such as meteorological data files, data formatting is applied on each row of the CSV files during the ingestion phase.

5.4 Data Access Use Case

In this section, we present a possible use case for the data ingested in our data lake architecture. Our use case is a monitoring dashboard for the different buildings of the university. Thus we are able to follow-up of the systems and infrastructures in the different buildings of the university. The sensor data can be used to monitor the ambient rooms inside the buildings. At the same time, we can follow the meteorological data collected by Météo France. By crossing both data, it is possible to investigate the possible correlations between these different data. Moreover, it is possible to detect malfunctions or failures of electrical or heating systems in a short time. The comparison of these data can allow us to distinguish a possible flooding from an open window in an equipped room. In Figure 6, we show the results of this use case in which we employ the InfluxDB as a web graphical visualization and dashboard creation tools. This tool allows us to visualize in real time via a native data refreshment system and allows us to follow in real time the evolution of the data. From the same dataset, it is possible to set up several different dashboards for different purposes. Thanks to InfluxDB, it is also possible to set up aggregation functions on the fly and allow processing directly in the dashboard without having to modify the data. Unit changes can be performed as in the example. It could be possible to apply more complex statistical analysis functions and to simply implement tools belonging to the field of statistical process control for the implementation of alerts by setting up control charts.

5.5 Metadata Management System

To facilitate the searching and use of metadata, we have implemented a metadata management web-application. The application has an ergonomic user-oriented interface through which users can easily find existing datasets, processes and analyses to find possibly useful information for data analytics. The front-end of application is developed with HTML/CSS and JavaScript. The back-end is developed with an API NeoViz to visualize the metadata stored in a Neo4j database.

To illustrate the application, we ingested real-time data from connected objects and batch datasets (CSV files) which are the two types of data used for NeOCampus. With the application, users can easily search existing datasets, processes or analysis by typing tags. Moreover, with the interface of the application, users can discover different aspects of information, such as basic element characteristics, lineage information and relationships among datasets. For example, when a user searches for humidity datasets, he can find that there is a dataset 190 which concerns humidity (see top image in Fig. 7). If he wants to know how the dataset is ingested, he can switch to the *Lineage* tab so that he finds out that a dataset source passed a real time process and then ingested into the dataset 190. Moreover, the lineage tab is available for batch data life cycle too, for instance, if users find a batch dataset, he can see a data source is

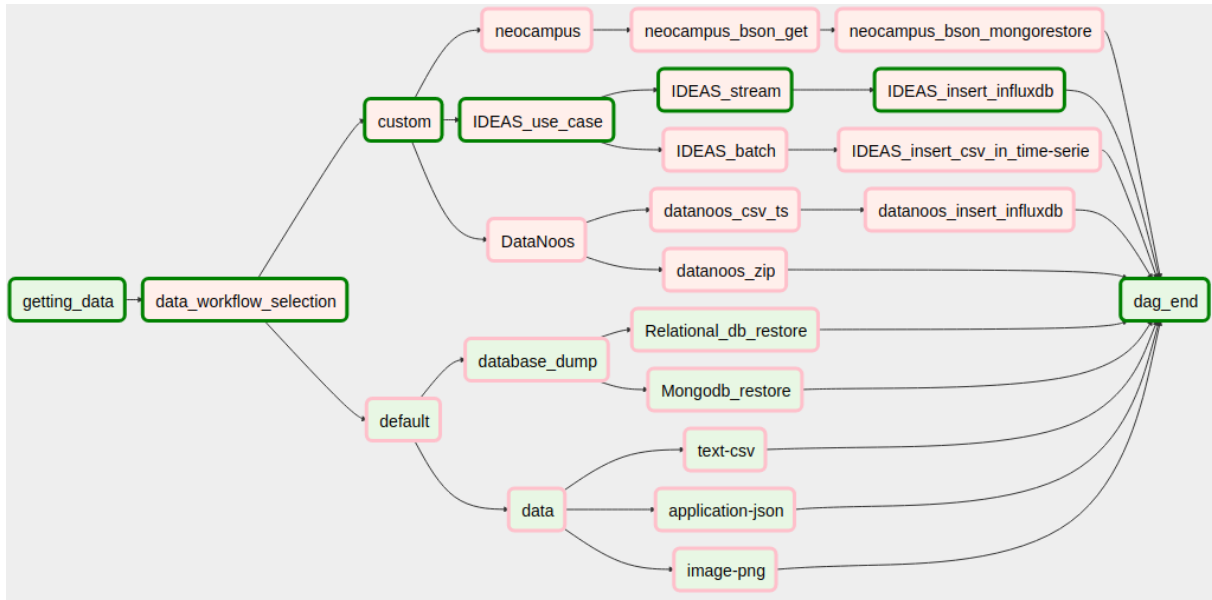


Figure 5: Different workflow pipelines visualization

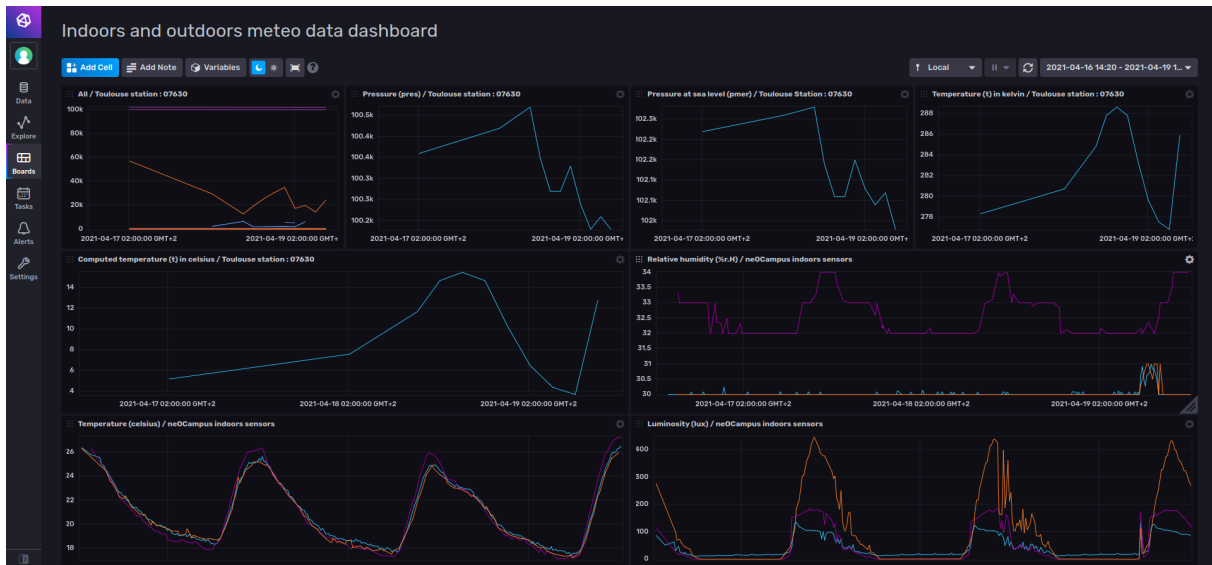


Figure 6: Indoors and Outdoors Monitoring

ingested directly into the data lake without any real time processing (see button image in Fig. 7).

6 CONCLUSIONS

This paper fall within the context of big data analytics with varied data. It includes batch data with large volume, stream data with high velocity or IoT sensor readings with great variety. Our solution addresses the issues raised by this new traffic created by connected objects while maintaining the primary applications of

big data analytics. From a theoretical point of view, with a functional architecture in four zones enriched with a metamodel that adapts to any type of data, as well as from a technical point of view, with an architecture designed to take advantage of the genericity of object-oriented storage. Our solution offers a solution that adapts to any type of data, in terms of volume or type. More than the ability to manage any type of data, it becomes possible to cross data and address the challenge of silo architecture and development in IoT.

We have implemented our solution through a concrete case (neOCampus operation) allowing the management of (i) data flows

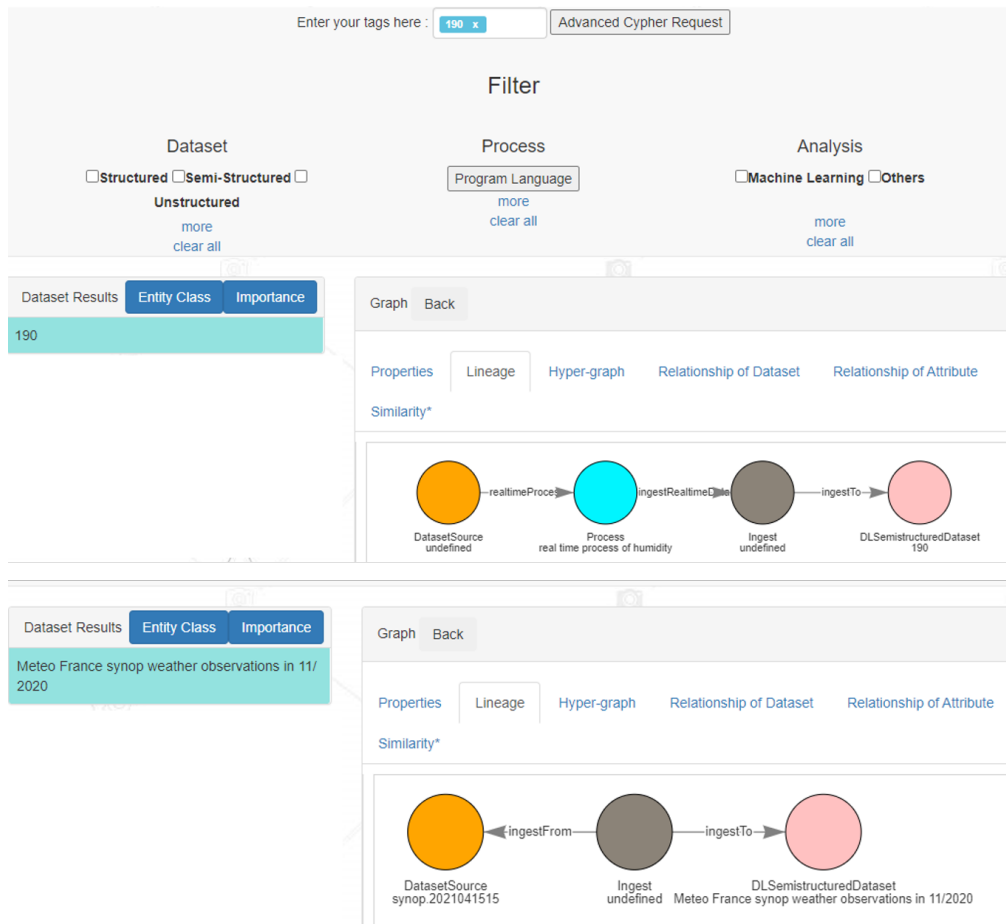


Figure 7: Searching results in the metadata management web application

(45 messages per minute) and (ii) datasets containing data from another context on about fifty weather stations on the whole French territory. The future work on this architecture will focus on 3 points: (i) security and the use of virtualization for multi-level user space separation and automatic deployment, (ii) working and processing tools, including the integration of tools for the analysis of data streams with temporal constraints and (iii) a strong integration of data semantics through metadata management.

REFERENCES

- [1] Ayman Alserafi, Alberto Abelló, Oscar Romero, and Toon Calders. 2016. Towards information profiling: data lake content metadata management. In *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)*. IEEE, 178–185.
- [2] Fabián Constante Nicolalde, Fernando Silva, Boris Herrera, and António Pereira. 2018. Big Data Analytics in IOT: Challenges, Open Research Issues and Tools. In *Trends and Advances in Information Systems and Technologies*, Álvaro Rocha, Hojjat Adeli, Luís Paulo Reis, and Sandra Costanzo (Eds.). Springer International Publishing, Cham, 775–788.
- [3] James Dixon. 2010. Pentaho, Hadoop, and Data Lakes. <https://jamesdixon.wordpress.com/2010/10/14/pentaho-hadoop-and-data-lakes/>
- [4] Mouzhi Ge, Hind Bangui, and Barbora Buhnova. 2018. Big data for internet of things: a survey. *Future generation computer systems* 87 (2018), 601–614.
- [5] Taiwo Kolajo, Olawande Daramola, and Ayodele Adebisi. 2019. Big data stream analysis: a systematic literature review. *Journal of Big Data* 6, 1 (2019), 1–30.
- [6] Ruoran Liu, Haruna Isah, and Farhana Zulkernine. 2020. A Big Data Lake for Multilevel Streaming Analytics. In *2020 1st International Conference on Big Data Analytics and Practices (IBDAP)*. IEEE, 1–6.
- [7] Mohsen Marjani, Fariza Nasaruddin, Abdullah Gani, Ahmad Karim, Ibrahim Abaker Targio Hashem, Aisha Siddiqua, and Ibrar Yaqoob. 2017. Big IoT Data Analytics: Architecture, Opportunities, and Open Research Challenges. *IEEE Access* 5 (2017), 5247–5261. <http://dblp.uni-trier.de/db/journals/access/access5.html#MarjaniNGKHSY17>
- [8] Imen Megdiche, Franck Ravat, and Yan Zhao. 2021. Metadata Management on Data Processing in Data Lakes. In *SOFSEM 2021: Theory and Practice of Computer Science - 47th International Conference on Current Trends in Theory and Practice of Computer Science, SOFSEM 2021, Bolzano-Bozen, Italy, January 25-29, 2021, Proceedings (Lecture Notes in Computer Science, Vol. 12607)*. Springer, 553–562.
- [9] Hassan Mehmood, Ekaterina Gilman, Marta Cortes, Panos Kostakos, Andrew Byrne, Katerina Valta, Stavros Tekes, and Jukka Riekkki. 2019. Implementing big data lake for heterogeneous data sources. In *2019 IEEE 35th international conference on data engineering workshops (icdew)*. IEEE, 37–44.
- [10] Amr A. Munshi and Yasser Abdel-Rady I. Mohamed. 2018. Data Lake Lambda Architecture for Smart Grids Big Data Analytics. *IEEE Access* 6 (2018), 40463–40471. <https://doi.org/10.1109/ACCESS.2018.2858256>
- [11] Franck Ravat and Yan Zhao. 2019. Data Lakes: Trends and Perspectives. In *Database and Expert Systems Applications - 30th International Conference, DEXA, Lecture Notes in Computer Science*. Springer International Publishing, 304–313.
- [12] Victoria Rubin and Tatiana Lukoianova. 2013. Veracity roadmap: Is big data objective, truthful and credible? *Advances in Classification Research Online* 24, 1 (2013), 4.
- [13] Shabnam Shadroo and Amir Masoud Rahmani. 2018. Systematic survey of big data and data mining in internet of things. *Computer Networks* 139 (2018), 19–47.
- [14] Yan Zhao, Imen Megdiche, and Franck Ravat. 2021. Analysis-oriented Metadata for Data Lakes. In *25th International Database Engineering Applications Symposium (IDEAS 2021)*.

COVID-19 Concerns in US: Topic Detection in Twitter

Carmela Comito

Nat. Research Council of Italy (CNR)

Institute for High Performance Computing and Networking (ICAR)

Rende, Italy

carmela.comito@icar.cnr.it

ABSTRACT

COVID-19 pandemic is affecting the lives of the citizens worldwide. Epidemiologists, policy makers and clinicians need to understand public concerns and sentiment to make informed decisions and adopt preventive and corrective measures to avoid critical situations. In the last few years, social media become a tool for spreading the news, discussing ideas and comments on world events. In this context, social media plays a key role since represents one of the main source to extract insight into public opinion and sentiment. In particular, Twitter has been already recognized as an important source of health-related information, given the amount of news, opinions and information that is shared by both citizens and official sources. However, it is a challenging issue identifying interesting and useful content from large and noisy text-streams. The study proposed in the paper aims to extract insight from Twitter by detecting the most discussed topics regarding COVID-19. The proposed approach combines peak detection and clustering techniques. Tweets features are first modeled as time series. After that, peaks are detected from the time series, and peaks of textual features are clustered based on the co-occurrence in the tweets. Results, performed over real-world datasets of tweets related to COVID-19 in US, show that the proposed approach is able to accurately detect several relevant topics of interest, spanning from health status and symptoms, to government policy, economic crisis, COVID-19-related updates, prevention, vaccines and treatments.

CCS CONCEPTS

• **Information systems** → **Social networking sites**; **Clustering**; **Data extraction and integration**.

KEYWORDS

COVID-19, Social Media Data, Topic Modeling

ACM Reference Format:

Carmela Comito. 2021. COVID-19 Concerns in US: Topic Detection in Twitter. In *25th International Database Engineering & Applications Symposium (IDEAS 2021)*, July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 8 pages. <https://doi.org/10.1145/3472163.3472169>

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Association for Computing Machinery.

ACM ISBN 978-1-4503-8991-4/21/07...\$15.00

<https://doi.org/10.1145/3472163.3472169>

1 INTRODUCTION

Twitter has long been used by the research community as a means to understand dynamics observable in online social networks, from information dissemination to the prevalence and influence of bots and misinformation. More importantly during the current COVID-19 pandemic, Twitter provides researchers the ability to study the role social media plays in the global health crisis.

Since early days of COVID-19, several papers exploiting social media data to address the epidemics-related issues have been proposed. The research activities focused on different topics: from the analysis of people reactions and the spread of COVID-19 [1, 15, 19, 20], to the search for conspiracy theories [9, 21] and the identification of misinformation propagation [11, 23]. A certain number of research works are also devoted on the detection of trending topics of discussion about COVID-19 [1, 7, 10, 16, 24].

The approaches in literature aiming at detecting COVID-19 topics from social media use the well know LDA topic modeling algorithm. However, traditional approaches to topic detection like LDA are not effective in a dynamic context like social media [3], where word vocabulary dynamically changes over time. In contrast, LDA data structures used to represent the textual content of tweets are fixed in advance, as the size of word vocabulary, the set of terms used, and the number of topics produced.

In the paper is described an alternative method able to detect COVID-19 topics on social media conversations exploiting the spatial-temporal features of the geo-tagged posts. Both numeric and textual features are extracted from the posts and their temporal evolution is monitored along the time to identify bursts in their values. The method proposed consists of a two-phase approach that first detects peaks in the time series associated to the spatio-temporal features of the tweets, and then, clusters the textual features (either hashtags or words) exhibiting peaks within the same timestamp. The clustering approach is based on co-occurrence of the textual features in the tweets. Moreover, by performing clustering on the textual features, we separate groups of words and topics that are perceived as more relevant for the COVID-19 debate. Debates range from comparisons to other viruses, health status and symptoms, to government policy, economic crisis, while the largest volume of interaction is related to the lockdown and the other countermeasure and restrictions adopted to fight the pandemics.

The rest of the paper is organized as follows. Section 2 overviews related work. Section 3 describes the target real-world dataset of tweets posted in US. Section 4 formulates the problem, introducing the key aspects of the approach and the topic detection algorithm. The results of the evaluation performed over the real-world datasets of tweets are shown in Section 5. Section 6 concludes the paper.

2 RELATED WORK

Since February 2020, an increasing number of studies focusing on the analysis of social media data related to the COVID-19 pandemics have been proposed. The majority of such works collect large-scale datasets and share them publicly to enable further research. Some datasets are restricted to a single language such as English [12, 13], while some others contain multiple languages [4, 7]. The dataset differentiate also for the collection periods. Among all, [4] stands out as one of the long-running collections with the largest amount of tweets (i.e., 250 million), also thanks to its multilingual nature. However, these datasets mostly contain only the raw content obtained from Twitter except that in [12, 13] associate a sentiment score with each tweet. The dataset in [18] enriches the raw tweet content with additional geolocation information. With more than 524 million tweets, the dataset is more than twice as large as the largest dataset available to date.

Another relevant bunch of works is devoted to the analysis of human behavior and reactions to the spread of COVID-19 [1, 15, 19, 20]; others focus on the detection of conspiracy theories and social activism [9, 21]. In [11, 23] are proposed approaches on misinformation propagation and quantification related to COVID-19 using twitter.

In the last years there has been a significant research effort on detecting topics and events in social media. This research line received also great attention for studying COVID-19 data on social media. Among the techniques defined for traditional data, and often adapted for Twitter data, as reported in [3], the most representative method is *Latent Dirichlet Allocation (LDA)* [6]. *LDA* is a topic model that relates words and documents through latent topics. It associates with each document a probability distribution over topics, which are distributions over words. A document is represented with a set of terms, which constitute the observed variables of the model. One of the main drawback of *LDA* is that data structures used to represent the textual content of tweets are fixed in advance as the size of word vocabulary, the set of terms used, and the expected number of topics.

Direct application of traditional approaches to topic detection like *LDA* on Twitter streams, as pointed out in [3], may give low quality results, thus, many different methods have been proposed [2, 5, 25, 26]. Generally, the common approach is to extract different data features from social media streams and then summarize such features for topic/event detection tasks by exploiting the information over content, temporal, and social dimensions. The approach proposed in this paper follows this view by introducing a set of spatio-temporal features to summarize tweet volumes user dynamics and location popularity and a set of textual features that summarize tweets content. Specifically, the main differences between the proposed approach and the above works is in the summarization technique. In [2, 5, 25, 26], the textual content of tweets is represented with a traditional vector-space model, where the vector dimension is fixed in advance as the size of word vocabulary. In streaming scenarios, because word vocabulary dynamically changes over time, it is very computationally expensive to recalculate the inverse document frequency of TF-IDF. Differently, we deal with content evolving signature structure of cluster by either updating frequencies of already present terms, or including new

terms; since we refer to term frequencies as relative values, there is no need of recalibrating the vocabulary size.

By the best of our knowledge all the works so far proposed to detect COVID-19 topics from social media adopt the *LDA* approach, thus, all of them suffer of the just cited limitations. The main relevant drawback is that the number of topics should be fixed in advance and the data structure and size has also to be fixed in advance. Anyhow, in order to give a complete view of what has been done to identify topics related to the COVID from social media, the current state-of-the-art is briefly surveyed below.

In [16] authors use twitter data to explore and illustrate five different methods to analyze the topics, key terms and features, information dissemination and propagation, and network behavior during COVID-19. The authors use pattern matching and topic modeling using *LDA* in order to select twenty different topics on spreading of corona cases, healthcare workers, and personal protective equipment. Using various analysis the authors were able to detect only few high level topic trends. Alrazaq et al. [1] also performed topic modeling using word frequencies and *LDA* with the aim to identify the primary topics shared in the tweets related to the COVID-19.

Chen et al. [7] analyzed the frequency of 22 different keywords such as "Coronavirus", "Corona", "Wuhan", analyzed across 50 million tweets from January 22, 2020 to March 16, 2020. Thelwall [24] also published an analysis of topics for English-language tweets during the period March 10-29, 2020. Singh et al. [23] analyzed distribution of languages and propagation of myths.

Sharma et al. [22] implemented sentiment modeling to understand perception of public policy on intervention policies such as "socialdistancing" and "workfromhome". They also track topics and emerging hashtags and sentiments over countries. Again, for topic detection they used *LDA*.

Cinelli et al. [8] compared Twitter against other social media platforms Instagram, YouTube, Reddit and Gab to model information spread about the COVID-19. They analyzed engagement and interest in the COVID-19 topic and provide a differential assessment on the evolution of the discourse on a global scale for each platform and their users. We fit information spreading with epidemic models characterizing the basic reproduction numbers R_0 for each social media platform. Also in this case, for topic detection users applied standard *LDA* method.

In [10] is described CoronaVis, a web application allowing to track, collect and analyze tweets related to COVID-19 generated from the USA. The tool allows to visualize topic modeling, analyze user movement information, study subjectivity and to model the human emotions during the COVID-19 pandemic. Those analysis is updated in real-time. They also share a cleaned and processed dataset named CoronaVis Twitter dataset (focused on United States) available to the research community at <https://github.com/mykabir/COVID19>. Also in this work for topic modeling authors used *LDA* and python pyLDAvis package to find out the relevant topics and produce an interactive visualization.

3 DATASETS

Real-world datasets of tweets posted in the United States have been used for the experimental evaluation. The choice to use tweets

Table 1: Keywords used to filter COVID-19 tweets.

Keywords	
corona	pandemic
breathing	lockdown
outbreak	distancing
death	quarantine
covid-19	pneumonia
virus	drops
Sars-cov-2	stayhome
China	mask
Wuhan	positive

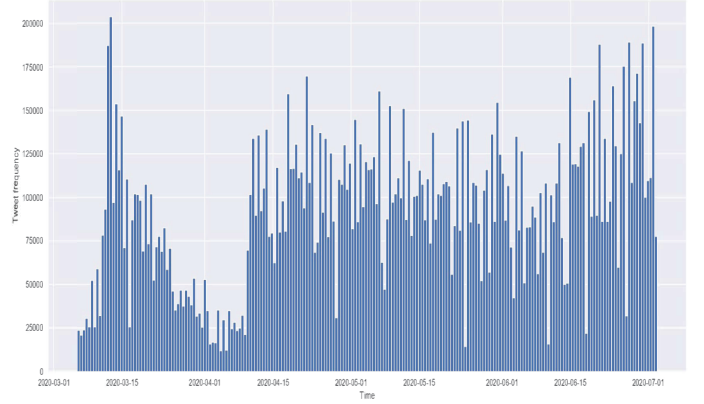
from US is mainly motivated by the large availability of data, making possible meaningful data analytics. In fact, United States has the most geotagged tweet mentions of COVID-19, followed by the United Kingdom.

Specifically, the dataset analysed is the one collected within the CoronaVis project [10] and accessible from the Github repository (<https://github.com/mykabir/COVID19>). Data were collected since March 5, 2020 using Twitter Streaming API2 and Tweepy3, and consists of over 200 million tweets related to COVID-19, which is about 1.3 terabytes of raw data saved as JSON files. The tweets have been posted by 30.070 unique users. Data is in the form (*TweetID*, *TweetText*, *UserLocation*, *UserType*). Some of the keywords used to filter COVID-19 related tweets are shown in Table 1.

Figure 1 depicts the temporal tweet frequency over the time. The graph shows that the daily tweet frequency is overall rather high but varies considerably from one day to another, spanning from 25,000 to 200,000, with some relevant peaks, e.g., on March 9th, end of April, mid May, June 21, end of June. The variability in the tweet frequency is also due to some gaps in the collected datasets caused by API and connectivity issues. Hence, in some of the date authors have fetched a fairly low amount of tweets compared to original number of the tweets on that day. To fill up those gaps we used another publicly available dataset, the GeoCOV19Tweets dataset [12].

The GeoCOV19Tweets dataset [12, 14, 18] consists of 675,104,398 tweets (available from the web site <https://iee-dataport.org/open-access/coronavirus-covid-19-geo-tagged-tweets-dataset>) and contains IDs and sentiment scores of the geo-tagged tweets related to the COVID-19 pandemic. The tweets have been collected by an on-going project deployed at <https://live.rhamsal.com.np>. The model monitors the real-time Twitter feed for coronavirus-related tweets using 90+ different keywords and hashtags that are commonly used while referencing the pandemic.

Data provided in the first dataset was already preprocessed and ready to be analysed. For what concerns the second dataset a pre-processing step has been performed: retweets were removed from the collected data together with punctuation, stop words, and non printable characters such as emojis from the tweets. Furthermore, various forms of the same word (eg, travels, traveling, and travelling) were lemmatized by converting them to the main word

**Figure 1: Tweets Frequency.**

(eg, travel) using the WordNetLemmatizer module of the Natural Language Toolkit Python library.

4 THE METHOD

4.1 Features modeling

The proposed method is articulated as a sequence of steps. As first step, a set of space-time features are extracted from social data so as to achieve two parallel temporal analysis: (i) the evolution of tweets, of the users who posted such tweets, and the locations from where the tweets have been posted; (ii) the evolution of tweets content through a textual analysis in which we extract the most relevant and frequent keywords in the text.

Accordingly, given a space-time window (e.g., a geographic region R and a timestamp t), the features are as follows. The number of tweets posted from R at time t is denoted as:

$$x_{TW_t} = |\mathcal{T}\mathcal{W}_t| \quad (1)$$

The number of users of R at t represents the distinct users who have posted at least one tweet in the region R at time t .

$$x_{U_t} = |\mathcal{U}_t| \quad (2)$$

The number of locations within R at t represents the number of distinct locations from where users have posted at least one tweet.

$$x_{L_t} = |\mathcal{L}_t| \quad (3)$$

The tweet entropy of R at t describes the distribution of tweets across the users, telling if users tend to tweet regularly in R at time t . For this purpose we use the Shannon Entropy:

$$x_{H_t} = - \sum_{u=1}^n f(t, u) \log f(t, u), \quad (4)$$

$f(t, u)$ is the user's proportion of tweets posted in R at time t and is defined as $f(t, u) = \frac{|\mathcal{T}\mathcal{W}_{t,u}|}{|\mathcal{T}\mathcal{W}_t|}$ where $\mathcal{T}\mathcal{W}_{t,u}$ is the set of tweets posted in R at time t by user u , while $\mathcal{T}\mathcal{W}_t$ is the set of tweets posted in R at time t .

The textual content of the tweets is represented by a set of features capturing the key elements of the text. Accordingly, the textual features like hashtags and words are extracted from the

tweets. To this purpose, a preliminary step is to extract and then preprocess the textual content of the tweets. The preprocessing of the tweets consists of the following steps:

1. Stemming and lemmatization. Different tokens might carry out similar information (e.g. tokenization and tokenizing). And we can avoid calculating similar information repeatedly by reducing all tokens to its base form using various stemming and lemmatization dictionaries.
2. Removing stop words and punctuation. Some tokens are less important than others. For instance, common words such as "the" might not be very helpful for revealing the essential characteristics of a text. So usually it is a good idea to eliminate stop words and punctuation marks before doing further analysis.
3. Computing term frequencies or tf-idf. For document clustering, one of the most common ways to generate features for a document is to calculate the term frequencies of all its tokens. Although not perfect, these frequencies can usually provide some clues about the topic of the document. And sometimes it is also useful to weight the term frequencies by the inverse document frequencies.

After the textual data preprocessing, the entropy values of the textual features are extracted and used as feature to achieve textual analysis. In the rest of the paper we refer to keywords or textual features interchangeably.

Since the textual features can be too much, to select the most relevant ones, among the most popular textual features (the ones with higher support), are selected the features with higher entropy.

The support of a textual feature tf posted in a geographic area R in t , is the number of tweets containing tf posted in R at time t .

$$s = |\mathcal{T}\mathcal{W}_{t,tf}| \quad (5)$$

The entropy of a textual feature tf posted from region R at timestamp t , describes the distribution of tweets containing tf across the users, telling if users tend to use regularly that textual feature in R at time t . For this purpose, the designed method uses the Shannon Entropy:

$$H_{tf_t} = - \sum_{u=1}^n f(t,u) \log f(t,u), \quad (6)$$

where $f(t,u)$ is the user's proportion of tweets containing tf posted in R at time t , and is defined as $f(t,u) = \frac{|\mathcal{T}\mathcal{W}_{t,u,tf}|}{|\mathcal{T}\mathcal{W}_{t,tf}|}$ where $\mathcal{T}\mathcal{W}_{t,u,tf}$ is the set of tweets containing tf posted in R at time t by user u , while $\mathcal{T}\mathcal{W}_{t,tf}$ is the support of the hashtag, that is the number of tweets containing tf .

Accordingly, for each keywords in the tweets posted in R at time t with support higher than a given threshold, we build the time series of its entropy feature:

4.2 Topics extraction

This section describes the approach proposed to identify topics from the tweets by exploiting the features introduced earlier. Given a space-time window, the algorithm provides a snapshot of what is going on in the specified geographic location at the given timestamp. Both space and time granularity may vary from the broader detail level until getting to the finer one. For example, the space

granularity can be set to a state, city or the municipal districts of the city, to surroundings or specific areas, or even precise venues (i.e., public buildings, private houses, etc). In the same way the temporal coordinate ranges from months to weeks to days, hours and minutes. The proposed method is articulated as a sequence of steps. As first step, we extract the set of space-time features as describe in the above section.

As second step, for each feature a time series is built. Then, an ad-hoc peak detection algorithm analyzes the time series and uses a score function to find peaks. The identification of peaks is merely the first step in the process of topic detection. Content analysis of the tweets where the peaks occur is necessary to identify the specific topic. To this aim, we modeled the most significant textual features also as time series. After that a clustering algorithm will group the textual peaks to identify trending topics of discussion related to the COVID-19 outbreak.

The key steps of the algorithm are peaks identification and topic detection through clustering of textual peaks. As basic principle, each element in the time series that deviates from a baseline profile is considered a peak. The baseline profile models normal user behavior and it is function of the time. and is computed through a scoring mechanism based on a statistics measure.

Let S be a function which associates a score $S(x_i, T_X)$ to the element x_i of a given time-series T_X . A given point x_i in T_X is a peak if $S(x_i, T_X) > 0$. A data point in a time-series is a local peak if (a) it is a large and maximum value within a time window; the value need not necessarily be globally maximum in the entire time-series; and (b) it is isolated i.e., not too many points in the window have similar values. To the purpose of the experimental evaluation presented in this paper, the time window is 6 hours and the time series spans over 4 days, thus it consists of 16 timestamps. As we are interested in identifying topics in nearly real-time we consider only the previous temporal values in the time series. Clearly, the approach also applies in for retrospective analysis. In this case, also the values successive in time series can be considered.

For a given point x_i in T_X , the score function computes the distances of x_i from its k left neighbors. Such distances can be computed by using a proper statistics measure. According to most of the literature work, the most suitable statistics for detecting peaks in time series are the median and the percentile. Accordingly, the median or percentile of the k preceding temporal values in T_X have to be computed.

To identify peaks the algorithm builds a scoring array S to compute the deviation of each element of the time series from the baseline profile obtained using the statistic s . The score is computed using the following score function:

$$S(x_t, T_i) = 1 - \frac{x_{baseline}}{x_t} \quad (7)$$

where $x_{baseline}$ is the baseline value for the time series T_i at time t , obtained applying the statistic s on the k points immediately preceding t in T_i . Notice that each time series T_i covers the time span from $t - k$ to t .

After peaks are identified, if the set of hashtags and words with textual peaks is not empty, the method activates the clustering algorithm that will be responsible for the topic detection.

To identify different topics emerging simultaneously in the target space-time window, we propose a clustering algorithm that exploits the co-occurrences of the keywords in the tweets. In particular, keywords exhibiting peaks, within the timestamp t , are grouped if they co-occur in a number of tweets based on a threshold value δ . The clustering algorithm allows to uniquely identify topics: each cluster $c \in C$ is associated to a topic and all the keywords in the cluster characterize the topic.

5 EXPERIMENTAL RESULTS

This section presents the preliminary results of the experimental evaluation performed on the dataset of geo-tagged tweets collected in US and presented in Section 3. The aim of the evaluation is to assess the effectiveness of the proposed topic detection approach and to identify the main COVID-19 related topics of discussion in US, in the period spanning from March 5 to June 21.

5.1 Performance Evaluation

The quality of the results obtained with the proposed method depends on the parameters of the two main phases of the approach: peaks identification, and clustering of the textual peaks. The key parameters of the peak detection step are the (i) statistics measure used for the baseline values and (ii) the set of features used to identify peaks. For what concerns the clustering, the parameters are the (i) support of the textual features and (ii) co-clustering threshold. The rest of the section evaluates the method performance with respect to such parameters.

To assess the performance of the peak detection step, the effectiveness of the different features have been evaluated, considering both median and percentile. The evaluation has been performed in terms of the following metrics:

- *Detected topics*. It is the percentage of detected topics:

$$DT = \frac{\text{Detected Topics}}{\text{Total Topics}}$$
- *Topic Rate*. It is the percentage of peaks of numeric features that become topics:

$$TR = \frac{\text{Detected Topics}}{\text{Total Peaks}}$$
- *Distinct Topic Rate*. It is the percentage of unique topics out to the total number of topics detected:

$$DTR = \frac{\text{Unique Topics detected}}{\text{Detected Topics}}$$

Results are shown in Figure 2. Several peaks of varying intensity and frequency have been identified for both numeric and textual features. However, not all the peaks correspond to a topic as the proposed approach identifies a topic through the clustering algorithm when both numeric and textual peaks are detected.

In general, the number of tweets (TW) is the feature producing the higher number of peaks; however, many of such peaks do not correspond to any topics, therefore the topic rate is the lowest. For what concerns the number of users feature (U), it produces also a relevant number of peaks even if lower than the one using the number of tweets and, differently to that, many of them are associated to trending topics with a good topic rate and a reasonably good percentage of detected topics. By using the number of

locations (L) as feature, the peaks obtained are rather small but almost all the peaks are actually associated to a topic with a consequent high topic rate. However, by using the location only a small percentage of topics are detected. The last considered feature is the entropy of tweets (H). With such a feature the peaks detected are substantially lower than the ones identified with the number of tweets, and conversely to that, almost all the peaks are associated to topics with the higher topic rate and percentage of topics detected. The entropy resulted to be the most discriminative feature: significant and concurrent changes in the number of tweets and in the number of users, as modeled by the entropy, is sign of interesting and really popular topics.

For what concerns the performance achieved by the different statistic metrics used to obtain the baseline threshold, the percentile measure is the most selective and, for such a reason, it does not intercept low peaks (e.g., the ones associated to topics that were quite popular across temporal windows, and, thus exhibiting a number of peaks of similar intensity in a close temporal horizon). On the contrary, the median it is able to catch even low intensity peaks and for such a reason some of them do not actually correspond to any topics. Therefore, even if the percentage of detected topics is higher, the topic rate is lower than the one achieved when using the percentile.

Results show that the optimal choice is the use of the median as statistics measure and the entropy of tweets as feature, setting that will be used throughout the experimental study.

To identify the optimal values of the clustering parameters, Table 2 evaluates the clustering structure obtained with different values of the co-clustering threshold and support, in terms of percentage of detected topics, topic rate and distinct topic rate.

Support	Co-clustering threshold	DT(%)	TR(%)	DTR(%)
s=0.25	$\delta = 0.2$	25%	8%	7%
s=0.25	$\delta = 0.4$	40%	12%	15%
s=0.25	$\delta = 0.6$	63%	25%	28%
s=0.25	$\delta = 0.8$	60%	30%	26%
s=0.50	$\delta = 0.2$	34%	70%	20%
s=0.50	$\delta = 0.4$	58%	75%	43%
s=0.50	$\delta = 0.6$	80%	89%	95%
s=0.50	$\delta = 0.8$	62%	86%	66%
s=0.75	$\delta = 0.2$	43%	72%	30%
s=0.75	$\delta = 0.4$	66%	74%	55%
s=0.75	$\delta = 0.6$	75%	89%	72%
s=0.75	$\delta = 0.8$	72%	82%	73%

Table 2: Evaluation of the clustering structure w.r.t. clustering parameters.

Table shows that for small values of both support and co-clustering threshold the algorithm produces big clusters grouping together tweets that are poorly connected and, thus, discussing

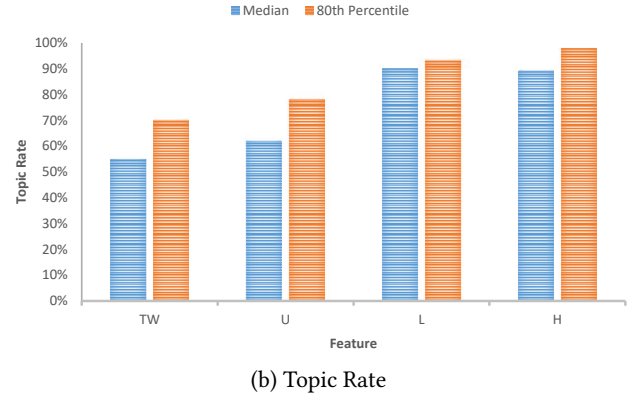
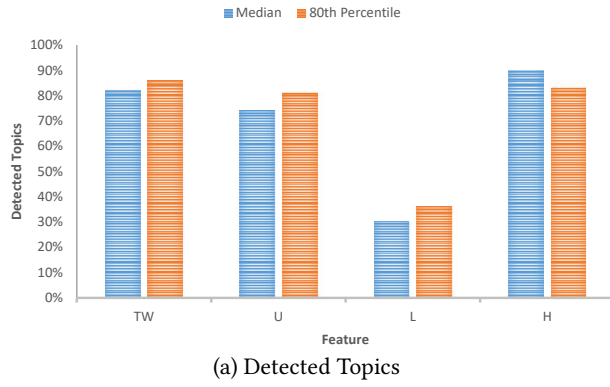


Figure 2: Performance of the different features w.r.t static metrics.

about different topics. This results in very bad values of the three evaluation metrics: low percentage of detected topics and a low topic rate. With the increasing of both support of the textual features and co-clustering threshold the algorithm groups better the tweets and its performance gradually improves. However, values of the clustering parameters too high deteriorate the algorithm performance. If the support of the textual features is too large, there is the risk of not considering smaller textual peaks corresponding to relevant topics, and consequently, the detection rate decreases. This could be, for example, the case of a topic described by several textual features that exhibits smaller peaks. On the other hand, for large values of the co-clustering threshold the algorithm does not group related tweets and creates a high number of small clusters with few tweets. Therefore, for the same topic several clusters are created. This results in a high percentage of clusters that became incorrectly topics, with a high percentage of duplicated topics and, thus, a small distinct topic rate. Summarizing, the optimal parameter setting resulted to be $s = 0.50$, $\delta = 0.6$.

5.2 COVID-19 related topics in US

In this section, we take a look at the content of the conversation taking place on Twitter about COVID-19 in US.

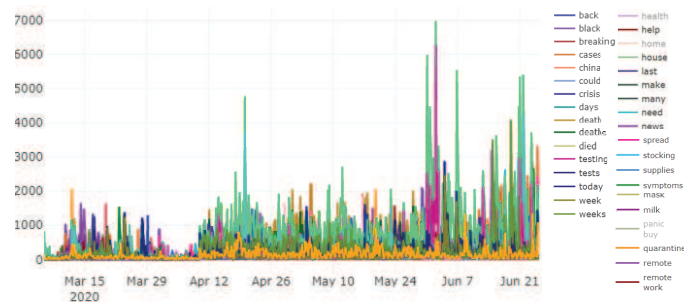


Figure 3: Overall textual peaks in US.

Figure 3 shows the most popular textual features all over US states. At the beginning of the epidemics the most used words were *corona*, *outbreak*, *spread*, *home*. Along the time, the words used in

the tweets change slightly: for example, on March 8 the most used were *pandemic*, *death*, *test*, *positive*; on March 19 news hashtags and words appeared like *#realdonaldtrump*, *week*, *stops*, *president*. On March 23 st, *positive*, *tested*, *patient*, *test*, *relief*. If we go further with time we find *death*, *even*, *test*, *health*, *positive*, *cases*.

The most affected US states till June 21, were California with 111.162 cases, Florida with 63,115 cases, New York with 48822 cases, Texas with 107232 cases, Washington with 50311 cases. Figures 4, 5, 6 show the textual features in California, Florida and New York state, respectively. As can be noted there are several peaks underlining the popularity of the textual features that varies with time. In particular, Figures 4-6 show that the states of California, Florida and New York present the same highest peaks along the same or very close temporal interval, and such peaks are recurrent throughout the period. This is the case of the textual peaks like *die*, *testing center*, *death*, *mask*, *vaccine*, *spread*.

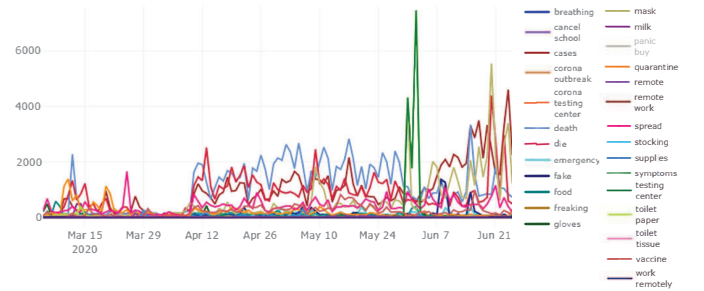


Figure 4: Overall textual peaks in California.

As an example, the situation in California is examined in more detail. Figure 4 shows that in California there are several textual peaks throughout the period. Peaks present different intensity and duration. Some of the textual features exhibit relevant peaks only the first time they appear like *corona outbreak*, *emergency*, *food*, *remote working*, *toilet papers*, *milk*; other textual features have recurrent peaks along the observed weeks like *death*, *die*, *spread*, *fake*, *cases*, *symptoms*, *vaccine*, *quarantine*, *breathing*. In such last case

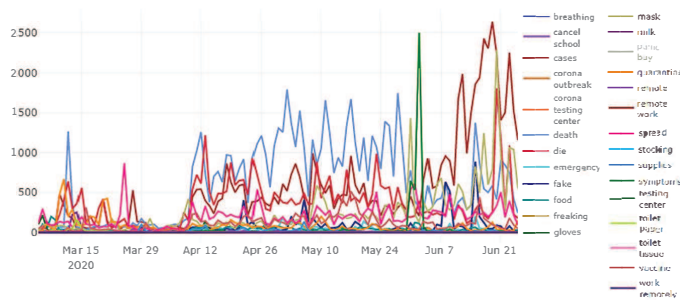


Figure 5: Overall textual peaks in Florida.

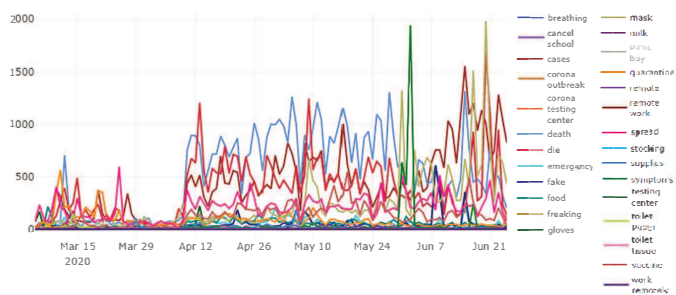


Figure 6: Overall textual peaks in NY.

there are few relevant peaks and the intensity of the textual features remains rather constant overall. Other features like *mask* or *testing center* present isolated peaks meaning that they become very popular in specific days.

Table 3 shows the top 12 topics detected in US in the target period, using as numeric feature the entropy of tweets and the entropy of the textual features; to obtain baseline values to extract peaks from the time-series the *median* is used.

The table highlights that the one of the most popular topic is related to the epidemics spreading with discussions about corona outbreak, deaths, COVID-19 cases, quarantine, breathing illness, emergency, mask, vaccines. Another relevant topic throughout the period is the topic *Death* related to the deaths caused by COVID-19 in the different US states. The topic *Travel issues* is related to the effects of COVID-19 on travel. Discussions were about flight cancellations, postponements, restrictions as well as travel warnings imposed by many countries due to the pandemic. In the topic *Panic buy* are grouped the tweets discussing about how people started to buy food and supplies due to lockdowns, and stay-at-home orders due to the COVID-19 pandemic, and how supermarkets and shops controlled and prevented panic buying. Another topic identified was the one related to *racism*. Specifically, users in most of the tweets reported the spreading of racist and xenophobic attacks about the disproportionate toll the Covid-19 pandemic was taking on communities of color. In the wake of the killing of George Floyd, the activism has intensified: doctors, epidemiologists, and nurses are increasingly abandoning their characteristic reticence in favor of direct advocacy. The topic *Medical supplies* concerns tweets about the importance of facial masks and gloves as prevention measures to reduce the outbreak and also their shortage in

several countries. Tweets about quarantining people infected or suspected to have COVID-19 are grouped in the topic *Quarantine*. At the same time with epidemics spreading the reactions of people were characterized by anxious and consequently a corresponding topic *Anxiety* emerged with words: quarantine, survivor, stress, worry, improve, post, anxiety, depression, family, physical.

6 CONCLUSIONS

Social networks have become a popular tool not only for information dissemination and individual opinion-making. Analyzing social network discussions can give us a perception of society and the world. In the current situation caused by COVID-19, understanding people awareness is extremely important. In this paper, we explored Twitter to understand the perception of people about COVID-19 and to assess what are the main topics of discussion related to the pandemics. To this purpose, is proposed an approach that detect topics by identifying peaks from tweet features. In particular, given a space-time window a set of features are extracted from the tweets, devised in: (i) numeric features, i.e. the number and the entropy of tweets, the number and the movements of distinct users; and (ii) textual features, i.e. the entropy of relevant hashtags. Each space-time feature is then modeled as a time series. Peaks of textual features co-occurring in a significant number of tweets are then associated to the same topic through a clustering algorithm based. Results, performed over real-world datasets of tweets, shown the feasibility of the proposed approach that is able to detect a large number of relevant COVID-19 topics.

REFERENCES

- [1] Alaa Abd-Alrazaq, Dari Alhuwail, Mowafa Househ, Mounir Hamdi, and Zubair Shah. 2020. Top Concerns of Tweepers During the COVID-19 Pandemic: Infoveillance Study. *J Med Internet Res* 22, 4 (21 Apr 2020), e19016.
- [2] Luca Maria Aiello, Georgios Petkos, Carlos Martin, David Corney, Symeon Papadopoulos, Ryan Skraba, Ayse Goker, Ioannis Kompatsiaris, and Alejandro Jaimes. 2013. Sensing Trending Topics in Twitter. *IEEE Transactions on Multimedia* 15, 6 (2013), 1268–1282.
- [3] Farzindar Atefeh and Wael Khreich. 2015. A Survey of Techniques for Event Detection in Twitter. *Comput. Intell.* 31, 1 (Feb. 2015), 132–164.
- [4] Juan M. Banda, Ramya Tekumalla, Guanyu Wang, Jingyuan Yu, Tuo Liu, Yuning Ding, and Gerardo Chowell. 2020. A large-scale COVID-19 Twitter chatter dataset for open scientific research – an international collaboration. arXiv:2004.03688 [cs.SI]
- [5] Hila Becker, Mor Naaman, and Luis Gravano. 2011. Beyond Trending Topics: Real-World Event Identification on Twitter. In *Proceedings of the Fifth International Conference on Weblogs and Social Media*.
- [6] David M. Blei, Andrew Y. Ng, Michael I. Jordan, and John Lafferty. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research* 3 (2003), 2003.
- [7] Emily Chen, Kristina Lerman, and Emilio Ferrara. 2020. Tracking Social Media Discourse About the COVID-19 Pandemic: Development of a Public Coronavirus Twitter Data Set. *JMIR Public Health and Surveillance* 6, 2 (May 2020), e19273. <https://doi.org/10.2196/19273>
- [8] Matteo Cinelli, Walter Quattrociocchi, Alessandro Galeazzi, Carlo Michele Valensise, Emanuele Brugnoli, Ana Lucia Schmidt, Paola Zola, Fabiana Zollo, and Antonio Scala. 2020. The COVID-19 Social Media Infodemic. arXiv:2003.05004 [cs.SI]
- [9] Emilio Ferrara. 2020. What types of COVID-19 conspiracies are populated by Twitter bots? *First Monday* (May 2020). <https://doi.org/10.5210/firstmonday.2516.10633>
- [10] Md. Yasin Kabir and Sanjay Madria. 2020. CoronaVis: A Real-time COVID-19 Tweets Analyzer. arXiv:2004.13932
- [11] Kraitam A El Alam MB Karam B Adib E Zarka J Traboulsi C Akl EW Baddour K Kouzy R, Abi Jaoude J. 2020. Coronavirus Goes Viral: Quantifying the COVID-19 Misinformation Epidemic on Twitter. *Cureus* 12, 3 (2020).
- [12] Rabindra Lamsal. [n.d.]. Coronavirus (COVID-19) Geo-tagged Tweets Dataset.
- [13] R. Lamsal. 2020. Corona Virus (COVID-19) Geolocation-based Sentiment Data. <https://doi.org/10.21227/fpsb-jz61>
- [14] Rabindra Lamsal. 2020. Coronavirus (COVID-19) Geo-tagged Tweets Dataset. <https://doi.org/10.21227/fpsb-jz61>

Table 3: Top 12 topics of discussion about COVID-19 in US.

Topics	Textual features
Covid-19 outbreak	2019-nCov, novel coronavirus, corona, Covid-19, outbreak, emergency
Lockdown	measure, lockdown, quarantine, prevention, stop
Death	dead, died, kill, death, sick, cough, patient, ill
Symptoms	symptoms, pain, pneumonia, fever, cough, sneeze, fatigue, headache, stomachache, lung, breath
Quarantine	home, close, lockdown, alone, family, inside, stay inside
Medical supplies	mask, disinfection, protection, shortage, gloves, gel
Anxiety	quarantine, survivor, stress, worry, anxiety, depression, family, physical
Vaccines	vaccine, immune, antibody, mouse, evaluate, vaccination, animal, protective
Remote Work	remote, work, working, remote working, working remotely, office, close, cancel school, home
Panic buy	panic buy, sold out, toilet, toilet tissue, food, meat, milk, water store, food, bought, gloves
Racism	covid, southern workers, fightback, organize nurses, blacklivesmatter, massnurses, racism, black people
Travel issues	travel, flying, flight, cruise, ticket, cancel, cancelled, fear

- [15] Lifang Li, Qingpeng Zhang, Xiao Wang, Jun Zhang, Tao Wang, Tian Lu Gao, Wei Duan, Kelvin Kam Fai Tsoi, and Fei Yue Wang. 2020. Characterizing the Propagation of Situational Information in Social Media during COVID-19 Epidemic: A Case Study on Weibo. *IEEE Transactions on Computational Social Systems* 7, 2 (1 April 2020). <https://doi.org/10.1109/TCSS.2020.2980007>
- [16] Catherine Ordun, Sanjay Purushotham, and Edward Raff. 2020. Exploratory Analysis of Covid-19 Tweets using Topic Modeling, UMAP, and DiGraphs. arXiv:2005.03082 [cs.SI]
- [17] Girish K. Palshikar. 2009. Simple Algorithms for Peak Detection in Time-Series. In *Proc. 1st Int. Conf. Advanced Data Analysis, Business Analytics and Intelligence*.
- [18] Umair Qazi, Muhammad Imran, and Ferda Ofli. 2020. GeoCoV19: A Dataset of Hundreds of Millions of Multilingual COVID-19 Tweets with Location Information. arXiv:2005.11177 [cs.SI]
- [19] Md Tahmid Rashid and Dong Wang. 2020. CovidSens: A Vision on Reliable Social Sensing for COVID-19. arXiv:2004.04565
- [20] Leonard Schild, Chen Ling, Jeremy Blackburn, Gianluca Stringhini, Yang Zhang, and Savvas Zannettou. 2020. "Go eat a bat, Chang!": An Early Look on the Emergence of Sinophobic Behavior on Web Communities in the Face of COVID-19. arXiv:2004.04046
- [21] Shadi Shahsavari, Pavan Holur, Timothy R. Tangherlini, and Vwani Roychowdhury. 2020. Conspiracy in the Time of Corona: Automatic detection of Covid-19 Conspiracy Theories in Social Media and the News. arXiv:2004.13783 [cs.CL]
- [22] Karishma Sharma, Sungyong Seo, Chuizheng Meng, Sirisha Rambhatla, and Yan Liu. 2020. COVID-19 on Social Media: Analyzing Misinformation in Twitter Conversations. arXiv:2003.12309 [cs.SI]
- [23] Lisa Singh, Shweta Bansal, Leticia Bode, Ceren Budak, Guangqing Chi, Kornraphop Kawintiranon, Colton Padden, Rebecca Vanarsdall, Emily Vraga, and Yanchen Wang. 2020. A first look at COVID-19 information and misinformation sharing on Twitter. arXiv:2003.13907 [cs.SI]
- [24] Mike Thelwall and Saheeda Thelwall. 2020. Retweeting for COVID-19: Consensus building, information sharing, dissent, and lockdown life. arXiv:2004.02793 [cs.DL]
- [25] Meng Wang, Kuiyuan Yang, Xian-Sheng Hua, and Hong-Jiang Zhang. 2010. Towards a relevant and diverse search of social images. *IEEE Transactions on Multimedia* 12, 8 (2010), 829–842.
- [26] Jie Yin, Andrew Lampert, Mark A. Cameron, Bella Robinson, and Robert Power. 2012. Using Social Media to Enhance Emergency Situation Awareness. *IEEE Intelligent Systems* 27, 6 (2012), 52–59.

IDEAS: the first quarter century

Bipin C. Desai*
BipinC.Desai@concordia.ca
Concordia University
Montreal, Canada

ABSTRACT

This year marks the silver anniversary of IDEAS. It has been an exciting quarter century to shepherd this meeting through good times and not so good ones. We have survived Ebola, MERS and SARS. Whereas the others were local, the Covid pandemic, which still rages, has forced us to move to an on-line version, but thanks to the participants and the dedicated program committee we have continued. This paper is a photographic journey through the years of IDEAS. Unfortunately we have not been able to have the images of all participants over the quarter century of IDEAS. Just a sampling of some of the fond moments during the social gatherings of the IDEAS family.

CCS CONCEPTS

• General and reference; • Software and its engineering; • Social and professional topics; • Applied computing; • Security and privacy;

KEYWORDS

Privacy, security, smartphone, contact tracking, big tech, surveillance

ACM Reference Format:

Bipin C. Desai. 2021. IDEAS: the first quarter century. In *25th International Database Engineering & Applications Symposium (IDEAS 2021), July 14–16, 2021, Montreal, QC, Canada*. ACM, New York, NY, USA, 15 pages. <https://doi.org/10.1145/3472163.3472168>

1 INTRODUCTION

Over the last 25 years, IDEAS has visited many cities, countries and three continents. We were able to hold IDEAS in Hong Kong in spite of SARS - luckily it was under control by the time of the meeting. Unlike the other epidemics, the current COVID pandemic is taking its toll on people's lives. We were forced to move IDEAS2020 which was to be hosted in S. Korea to an on-line version. We were hopeful that the incidence of epidemics, like EBOLA, MERS and SARS

*Corresponding Author

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, or to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Association for Computing Machinery.

ACM ISBN 978-1-4503-8991-4/21/07...\$15.00

<https://doi.org/10.1145/3472163.3472168>

would be short lived. We were hoping that, like the previous epidemics, COVID would be limited and there would not be further disruptions after the summer of 2020! Alas the pandemic has its own agenda and the first wave was followed by a more deadly second wave and rapidly by a third. The organizing committees seeing this and the slow progress in the vaccination efforts, reluctantly decided to move our silver jubilee event to on-line as well. In spite of this last minute change, we were glad to receive a large number of submissions. The current proceedings is an evidence of the high quality of work in database, and data analytics.

1.1 Our meetings

Following is the list of the IDEAS meetings over the past quarter of a century.

- IDEAS 1997: International Database Engineering and Applications Symposium, Aug. 25-27, 1997, Concordia University, Montreal, QC, Canada, Eds. Bipin.C. Desai, Barry Eaglestone: IEEE Computer Society 1997; ISBN: : 0-8186-8114-4; DOI: <https://doi.org/10.1109/IDEAS.1997>
- IDEAS 1998; 2nd International Database Engineering and Applications Symposium, July 8-10, 1998. University of Cardiff, Eds. Barry Eaglestone, Bipin C. Desai, Jianhua Shao: IEEE Computer Society 1998; ISBN: 0-8186-8307-4; DOI: <https://doi.org/10.1109/IDEAS.1998>
- IDEAS 1999: 3rd International Database Engineering and Applications Symposium, August 2-4, 1999, Concordia University, Montreal, Canada, Eds. Bipin C. Desai, Gösta Grahne: IEEE Computer Society 1999; ISBN: 0-7695-0265-2; DOI: <https://doi.org/10.1109/IDEAS.1999>
- IDEAS 2000: 4th International Database Engineering and Applications Symposium, IDEAS 2000, September 18-20, 2000, Keio University, Yokohoma, Japan, Eds: Bipin C. Desai, Yasushi Kiyoki, Motomichi Toyama: IEEE Computer Society 2000; ISBN: 0-7695-0789-1; DOI: <https://doi.org/10.1109/IDEAS.2000>

- IDEAS 2001: 5th International Database Engineering and Applications Symposium, July 16-18, 2001, Grenoble, France: Eds: Michel E. Adiba, Christine Collet, Bipin C. Desai: IEEE Computer Society 2001; ISBN: 0-7695-1140-6; DOI: <https://doi.org/10.1109/IDEAS.2001>
- IDEAS 2002: 6th International Database Engineering and Applications Symposium, July 17-19, 2002, Edmonton, Canada, General Chair: Bipin C. Desai, Eds: Mario A. Nascimento, M. Tamer Özsu, Osmar R. Zaiane: IEEE Computer Society 2002; ISBN: 0-7695-1638-6; DOI: [10.1109/IDEAS.2002](https://doi.org/10.1109/IDEAS.2002)
- IDEAS2003: 7th International Database Engineering and Applications Symposium, 16-18 July 2003, Hong Kong, China. Ed. Bipin C. Desai, Wilfred Ng: IEEE Computer Society 2003; ISBN: 0-7695-1981-4; DOI: <https://doi.org/10.1109/IDEAS.2003>
- IDEAS 2004: 8th International Database Engineering and Applications Symposium (IDEAS 2004), 7-9 July 2004, Coimbra, Portugal. Ed. Bipin C. Desai, Jorge Bernardino: IEEE Computer Society 2004; ISBN: 0-7695-2168-1; DOI: <https://doi.org/10.1109/IDEAS.2004>
- IDEAS 2004DH: IDEAS Workshop on Medical Information Systems: The Digital Hospital (IDEAS-DH'04), 1-3 Sept. 2004, Beijing, China – 2004 Ed: Bipin C. Desai; Keqin Rao; Baoluo Li; Fulin Zhang; ISBN: 0-7695-2289-0; DOI: <https://doi.org/10.1109/IDEADH.2004>
- IDEAS 2005: 8th International Database Engineering and Applications Symposium (IDEAS 2005), 25-27 July 2005, Montreal, Canada. Ed. Bipin C. Desai, Gottfried Vossen: IEEE Computer Society 2005; ISBN: 0-7695-2404-4; DOI: <https://doi.org/10.1109/IEEECONF10944.2005>
- IDEAS 2006: 10th International Database Engineering and Applications Symposium (IDEAS 2006), 11-14 December 2006, Delhi, India. Ed. Bipin C. Desai, Shyam K. Gupta: IEEE Computer Society 2006; ISBN: 0-7695-2577-6; DOI: <https://doi.org/10.1109/IEEECONF11732.2006>
- IDEAS 2007: 11th International Database Engineering and Applications Symposium (IDEAS 2007), September 6-8, 2007, Banff, Alberta, Canada. Ed. Bipin C. Desai, Ken Barker: IEEE Computer Society 2007, ISBN: 0-7695-2947-X, DOI: <https://doi.org/10.1109/IEEECONF13176.2007>
- IDEAS 2008: 12th International Database Engineering and Applications Symposium (IDEAS 2008), September 10-12, 2008, Coimbra, Portugal. General Chair: Bipin C. Desai, ACM International Conference Proceeding Series 299, ACM 2008, ISBN: : 978-1-60558-188-0, DOI: [10.1145/1451940](https://doi.org/10.1145/1451940)
- IDEAS 2009: 13th International Database Engineering and Applications Symposium (IDEAS 2009), September 16-18, 2009, Cetraro, Calabria, Italy. Ed. Bipin C. Desai, Domenico Saccà, Sergio Greco: ACM International Conference Proceeding Series, ACM 2009; ISBN: 978-1-60558-402-7; DOI: [10.1145/1620432](https://doi.org/10.1145/1620432)
- IDEAS 2010: 14th International Database Engineering and Applications Symposium (IDEAS 2010), August 16-18, 2010, Montreal, Quebec, Canada; General Chair: Bipin C. Desai, Program Chair: Jorge Bernardino: ACM International Conference Proceeding Series, ACM 2010; ISBN: 978-1-60558-900-8; DOI: [10.1145/1866480](https://doi.org/10.1145/1866480)
- IDEAS 2011: 15th International Database Engineering and Applications Symposium (IDEAS 2011), September 21 - 27, 2011, Lisbon, Portugal; Ed: Bipin C. Desai, Isabel F. Cruz, Jorge Bernardino, ACM International Conference Proceeding Series; ISBN: 978-1-4503-0627-0; DOI <https://doi.org/10.1145/2076623>
- IDEAS 2012: 16th International Database Engineering and Applications Symposium, IDEAS '12, Prague, Czech Republic, August 8-10, 2012. General Chair: Bipin C. Desai, Program Chairs: Jaroslav Pokorný, Jorge Bernardino ACM International Conference Proceeding Series, 2012; ISBN: 978-1-4503-1234-9; DOI <https://doi.org/10.1145/2351476>
- IDEAS 2013: 17th International Database Engineering and Applications Symposium, IDEAS '13, Barcelona,

Spain - October 09 - 11, 2013. Bipin C. Desai, Josep Lluís Larriba-Pey, Jorge Bernardino: ACM 2013; ISBN: 978-1-4503-2025-2; DOI: <https://doi.org/10.1145/2513591>

- IDEAS 2014: 18th International Database Engineering and Applications Symposium, IDEAS 2014, Porto, Portugal, July 7-9, 2014. General Chair: Bipin C. Desai: ACM 2014; ISBN: 978-1-4503-2627-8; DOI: <https://doi.org/10.1145/2628194>
- IDEAS 2015: 19th International Database Engineering and Applications Symposium, Yokohama, Japan, July 13-15, 2015. Eds. Bipin C. Desai, Motomichi Toyama: ISBN: 978-1-4503-3414-3; DOI: <https://doi.org/10.1145/2790755>
- IDEAS 2016: 20th International Database Engineering and Applications Symposium, IDEAS 2016, Montreal, QC, Canada, July 11-13, 2016. Eds. Evan Desai, Bipin C. Desai, Motomichi Toyama, Jorge Bernardino: ISBN: 978-1-4503-4118-9 DOI: <https://doi.org/10.1145/2938503>
- IDEAS 2017: 21st International Database Engineering and Applications Symposium, IDEAS 2017, Bristol, United Kingdom, July 12-14, 2017. Eds. Bipin C. Desai, Jun Hong, Richard McClatchey; ISBN: 978-1-4503-5220-8; DOI: <https://doi.org/10.1145/3105831>
- IDEAS 2018: 22nd International Database Engineering and Applications Symposium, IDEAS 2018, Villa San Giovanni, Italy, June 18-20, 2018. Ed. Bipin C. Desai, Sergio Flesca, Ester Zumpano, Elio Masciari, Luciano Caroprese; ISBN: 978-1-4503-6527-7; DOI: <https://doi.org/10.1145/3216122>
- IDEAS 2019: 23rd International Database Applications and Engineering Symposium, IDEAS 2019, Athens, Greece, June 10-12, 2019. Ed. Bipin C. Desai, Dimosthenis Anagnostopoulos, Yannis Manolopoulos, Mara Nikolaidou; ISBN: 978-1-4503-6249-8; DOI: <https://doi.org/10.1145/3331076>

- IDEAS 2020: 24th International Database Engineering and Applications Symposium, Seoul, Republic of Korea, August 12-14, 2020. Ed. Bipin C. Desai, Wan-Sup Cho: ACM 2020; ISBN: 978-1-4503-7503-0; DOI: <https://doi.org/10.1145/3410566>
- IDEAS 2021: 25th International Database Engineering and Applications Symposium, Montreal, Canada, July 14-16, 2021. General Chairs: Bipin C. Desai, Jeffrey Ullman Program Chairs: Richard McClatchey, Motomichi Toyoma; ISBN: 978-1-4503-8991-4; DOI: TBA

2 A PICTORIAL LOOK BACK

Over this quarter century, IDEAS had many participants; some have been graduate students or in the early phase of their career; others were more established. Many of these contributors have continued to participate in these annual meetings. Others are either retired or making plans! The torch has passed from these to new ones who keep joining this IDEAS family. The following photos have some fond memories to be shared!

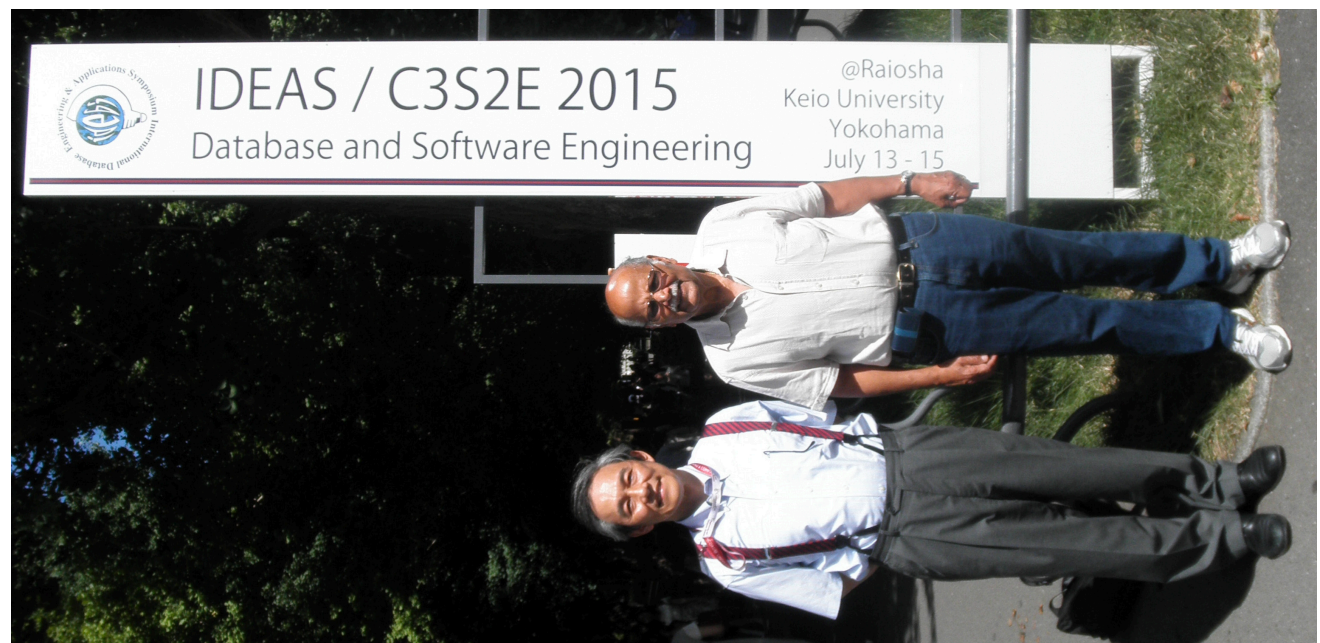




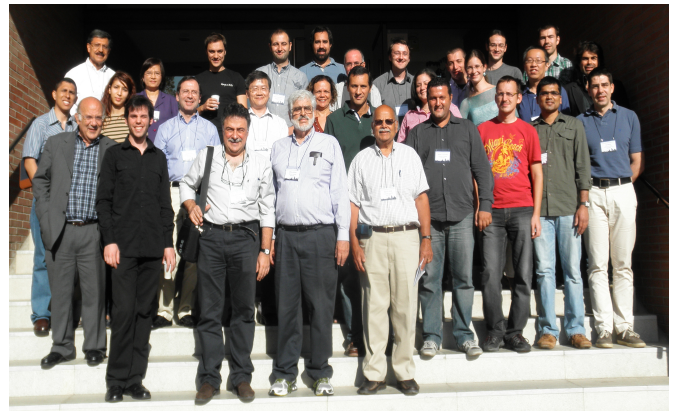
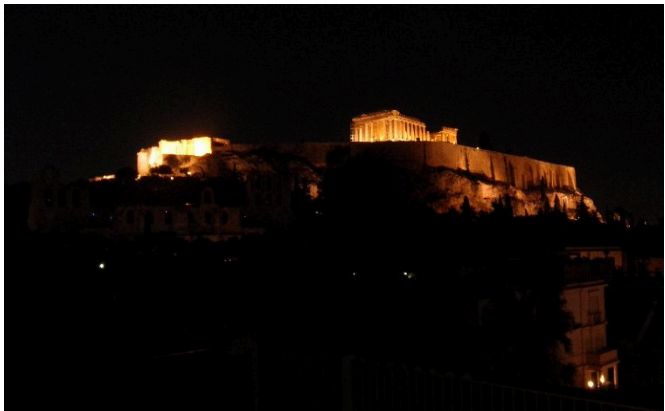


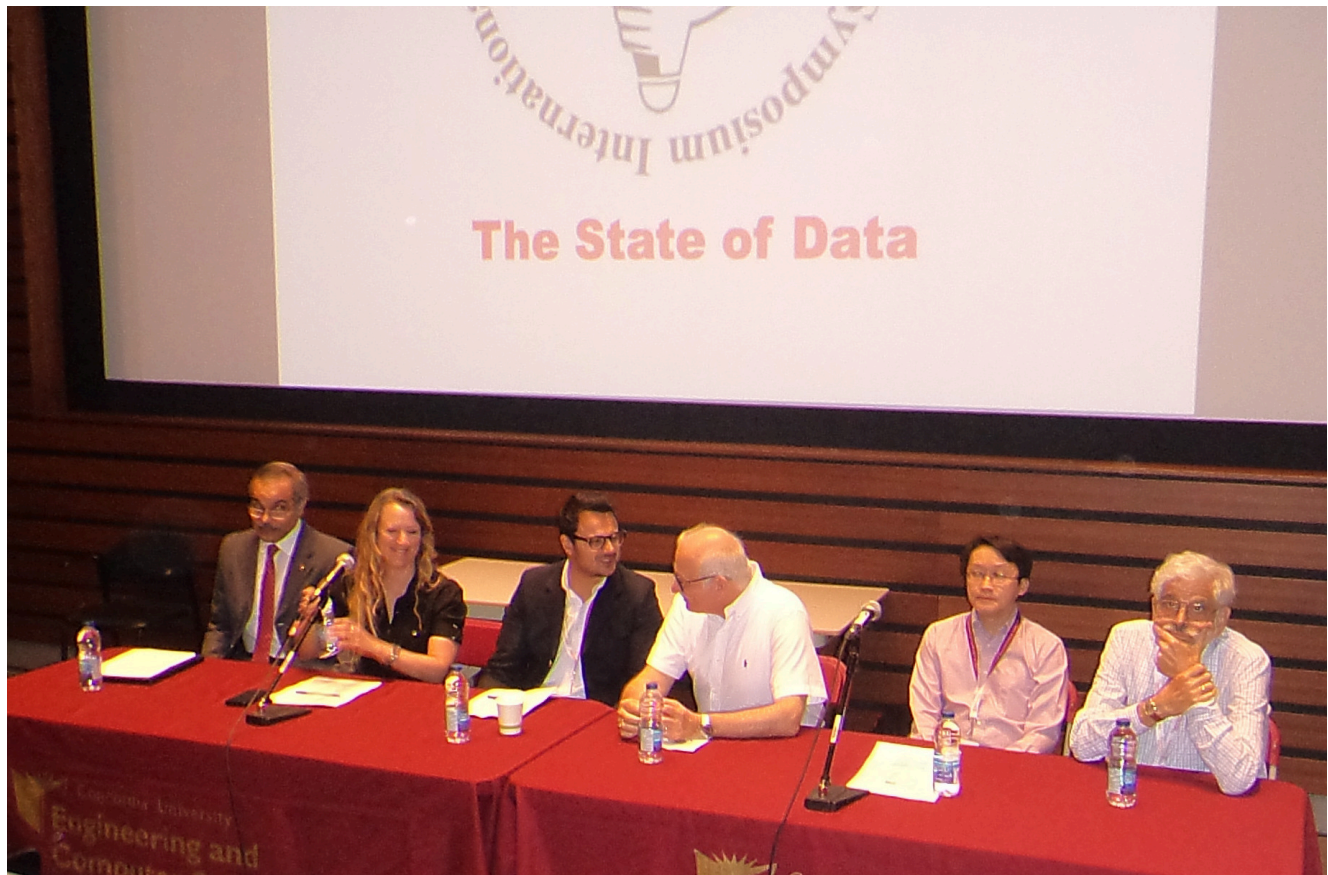




















Load Balanced Semantic Aware Distributed RDF Graph

Ami Pandat
Distributed Databases Group
DAIICT
Gandhinagar, India
202021002@daiict.ac.in

Nidhi Gupta
Distributed Databases Group
DAIICT
Gandhinagar, India
201601048@daiict.ac.in

Minal Bhise
Distributed Databases Group
DAIICT
Gandhinagar, India
minal_bhise@daiict.ac.in

ABSTRACT

Modern day application development requires efficient management of huge RDF data. The major approaches for RDF data management are Relational and Graph based techniques. As the relational approach suffers from query joins, we propose a semantic aware graph based partitioning method. The partitioned fragments are further allocated in a load balanced way. For efficient query processing, partial replication is implemented. It reduces Inter node Communication thereby accelerating queries on distributed RDF Graph. This approach has been demonstrated in two phases partitioning and Distribution of Linked Observation Data (LOD). The time complexity for partitioning and distribution of Load Balanced Semantic Aware RDF Graph (LBSD) is $O(n)$ where n is the number of triples which is demonstrated by linear increment in algorithm execution time (AET) for LOD data scaled from 1x to 5x. LBSD has been found to behave well till 4x. LBSD is compared with the state of the art relational and graph-based partitioning techniques. LBSD records 71% QET gain when averaged over all the four query types. For most frequent query types, Linear and Star, on an average 65% QET gain is recorded over original configuration for scaling experiments. The optimal replication level has been found to be 12% of original data.

CCS CONCEPTS

• Information systems → Graph-based database models.

KEYWORDS

Degree Centrality, Graph Partitioning, Load Balanced Distribution, LOD, Partial Replication, Semantic Aware Partial Replication

ACM Reference Format:

Ami Pandat, Nidhi Gupta, and Minal Bhise. 2021. Load Balanced Semantic Aware Distributed RDF Graph. In *25th International Database Engineering & Applications Symposium (IDEAS 2021), July 14–16, 2021, Montreal, QC, Canada*. ACM, New York, NY, USA, 7 pages. <https://doi.org/10.1145/3472163.3472167>

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada
© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-8991-4/21/07.
<https://doi.org/10.1145/3472163.3472167>

1 INTRODUCTION

A W3C standard, Resource Description Framework (RDF) is a foundation of semantic web and used to model web objects. An RDF dataset comprises triples in the form of (subject, property, and object). It can be readily comprehended as a graph, where subjects and objects are vertices joined by labeled relationships i.e., edge. It is however now being used in a broader context. Bio2RDF[4] data collection is used by biologists to store their experimental results in RDF triples to support structural queries and communicate among themselves. Similarly, DBpedia[5] extracts information from Wikipedia and stores it as RDF data. W3C offers a structured query language, SPARQL to retrieve and manage the RDF datasets. Finding an answer to the SPARQL query requires finding a match of the subquery graph in the entire RDF graph. As the RDF data is gaining acceptance widely, RDF dataset sizes are moving from a centralized system to distributed system.

There are two techniques for RDF data management: relational and graph-based. In the relational method, data is kept in the form of multiple tables. To find an answer to a query, one needs to extract that information from various tables by applying the join operation. On the other hand, in the graph-based technique data is represented in the form of vertices and edges. Semantic partitioning [23] is one of the graph partitioning technique, implemented for a centralized system using page-rank algorithms. To work towards building efficient partitioning and distribution algorithms, there are many state of the art available.

Some of the partitioning algorithms use the query workload to identify the parts of the RDF graph which are used frequently and keep these subgraphs at one site. While this approach works well for the systems in which the majority of queries follow the identified query patterns, it may not work as well in the systems where new queries do not correlate with the existing workload. The configuring system that doesn't use workload information is desirable. Instead, if we use the semantics of RDF to partition the data, algorithm execution time would be much lower and query execution time for new queries would either be the same or better than the workload aware methods. Semanticity of RDF data refers to the format of triples in a Turtle or N-Quad RDF file. This triple data file can directly be used for partition and distribution using the fact that the edge is denoted by the equivalence of subject and object in two triples. Using this structure of triples, one can directly work on complexities that are based on the number of triples in a file.

Reviewing such kind of aspects and agendas available in graph-based techniques, this research is designed to develop algorithms to partition data using semantic relation between vertices and distribute among several nodes. Load Balanced Semantic Aware Graph

(LBSD) uses semantic partitioning, for the initial phase of partitioning. The system partitions data and makes clusters. At that point, it will disseminate applicable bunches (by semantic connection) among the given number of hubs. The fundamental reason to segment RDF information is to answer inquiries effectively in a lesser measure of time. To reduce inter node communication (INC) in distributed environment, partial replication [13] of data has been done. It is demonstrated by deciding how much amount of data should be replicated over every node to reduce INC.

The rest of the paper is organized as follows: in the next section, we discuss related work regarding this research. In Section 3, we discuss the methodology used to implement this work. Section 4 describes the details of experiments and evaluation parameters. In Section 5, we discuss the results and comparison of the system with the state of the art work, and then finally Section 6 states the conclusion.

2 RELATED WORK

The present approaches for handling the huge RDF data can be classified into two categories; Relational and Graph-based approaches.

2.1 Relational Approaches

RDF triples can naturally be implemented as a single table with three columns specifically subject, predicate object. This table can have millions of triples. This approach aims to utilize the well developed techniques in conventional relational techniques for query processing, and storage of data. Research in relational techniques deals with the partitioning of RDF tables in such a way that there is a substantial decrease in the number of joins while answering a query.

Property tables approach utilizes the repeated appearances of patterns and stores correlated properties in the same table. Class property table and clustered property table are two techniques in which the former defines various tables that contain a particular property value while the latter defines a table for a particular subject [1].

DWAHP [19] is the relational technique partitions the data using workload aware approach using n-hops property reachability matrix. Clustering of Relational data in distributed databases for medical information is discussed in [21] which is also similar kind of the state of the art work for relational systems. It uses Horizontal Fragmentation for the implementation. This technique is implemented for relational approach and this research LBSD discusses the same for graph-based approach.

The relational approach for SPARQL-based query known as Direct relational mappings in which a SPARQL query can be translated to SQL query for given data in the form of the triple [24]. Another technique is single table extensive indexing which is used to develop native storage systems that allow extensive indexing of the triple table. e.g. Hexastore and 3X [18]. SIVP [22] proposes Structure Indexed Vertical Partitioning which combines structure indexing and vertical partitioning to store and query RDF data for interactive semantic web applications. It presents five metrics to measure and analyze the performance of the SIVP store. SIVP is better than vertical partitioning provided the extra time needed in SIVP, which consists of lookup time and merge time, is compensated by frequency. Above all are relational approaches which closely relate to LBSD in some or other way.

2.2 Graph Based Approaches

The graph-based technique eliminates query joins. It maintains the original representation of the RDF data and implements the semantics of RDF queries but it scales poorly. Several recent works deal with RDF graph partitioning. gStore [24] is a system designed to exploit the natural graphical structure of RDF triples. It also executes the queries using the subgraph matching approach. The Graph-based technique, Adaptive partitioning and Replication (APR) [2] works to partition query graphs using Workload information, and then it decides the benefit level to make a certain decision that how much data should be replicated in the given graph.

Another approach is UniAdapt [3]. This technique proposes a unified optimization approach that enables a distributed RDF Triple Store to adapt its RDF Storage layer by focusing on replication as well as main memory indexes. The final objective for this approach to decrease future query execution time. METIS [14] is one of the popular baselines for multiple works. [10] [9] [7]. APR [2] first partitions the graph using METIS and then uses a global query graph made using workload for replication. To handle query processing efficiently in distributed environment, edge-cut novel approach using two strategies 1. Overpartitioned minimal edge-cut cover. 2. Our novel molecule hash cover [12]. Its analysis substantiates hypothesis by explaining the causes for their good performance. Both strategies reduce query execution time on our set of test queries (between 5% and 98%).

The other approach uses the semantic properties of RDF data and proposes a Page Rank inspired algorithm to cluster the RDF data [23]. This approach is implemented for centralized system whereas proposed technique LBSD inspired by the same but works for distributed systems. One more recent approach [20] uses the frequency of query patterns to partition the graph and proposes three methods of fragmentation. Other than relational and graph-based approach there are approaches which deal with index, dataset formats and storage structure. While partitioning and distributing data, the index of data fed to the system and the format of data are also key features. Several partitioning techniques available to handle query workload for static partitioning, which turns into the result that 40% query remains unanswered [11]. These types of shortcomings are resolved in [8], which handles dynamic ranged partitioning using workload information.

Present work in graph based RDF Data management needs workload information for partitioning and distribution and does not address load balancing issue. The present research work in semantic aware partitioning is demonstrated on centralized system only. These limitations of present work motivates this research to implement LBSD to support semantic aware partitioning in a distributed environment. LBSD has two phases: 1. Semantic aware partitioning 2. Distribution using partial replication. It aims to reduce the communication cost during SPARQL query processing. It adaptively maintains some frequent access patterns (FAPs) to reflect the characteristics of the workload while ensuring the data integrity and approximation ratio. To reduce INC, data should be replicated among all local nodes by its semantic relation and for that, a partial replication technique can be used. The partial replication technique

Algorithm 1: Extraction of popular Nodes**Input:** RDF Triples, no. of fragments**Output:** list of frequent subjects

```

1 Function EXTRACTION(RDF Triples, k fragments):
2   hashmap h with key = subject and value = frequency
3   for each triple in triples do
4     | h(subject)++
5   sort the hashmap by value
6   return top k subjects

```

decides the replication level using certain criteria and replicates the vertices which are most frequently used or most relevant. LBSD uses the similar technique for graph based approach using Centrality concept.

3 RESEARCH METHODOLOGY

The LBSD aims to distribute RDF data using graph based approach over available nodes to reduce inter-node communication (INC). The methodology divided into two phases. First Phase is Semantic aware Partitioning of RDF Data which consists of two algorithms. Algorithm 1 is used for extraction of popular nodes and algorithm 2 is used for partitioning. The Second Phase is Distribution of RDF Data, includes algorithm 3 and algorithm 4 for distribution and replication respectively. Figure 1 depicts the same. As shown in Figure 1, first available datasets of RDF Data will be transformed from CSV to ttl data file to set input into graph-based tools. The .ttl data file will be as tripled data which then will be fragmented and distributed in upcoming phases.

3.1 Partitioning of RDF Graph

Our aim for designing a fragmentation algorithm is to reduce INC, especially for linear and star queries. For example, social media data may have frequent star queries to get the friends of a person. RDF data has an advantage because it represents the data in the form of triple $\langle \text{subject}, \text{predicate}, \text{object} \rangle$. First, we need to find out the subjects which have many outgoing degrees. If we put these popular subjects at different nodes, then we can get rid of INC for star queries.

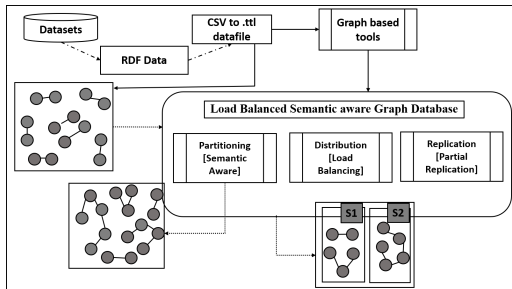


Figure 1: Block diagram for LBSD

Each of these popular subjects is known as the master nodes. We can do this by sending the incoming query to the node in which the

corresponding subject resides. Suppose there are k fragments, then we need to find k subjects with the highest number of outgoing links. Algorithm 1 lists out the steps needed to do that. Since algorithm 1 goes through all the triples only once, it has a linear time complexity $O(n)$ where n is the number of triples. Sorting HashMap would take $O(m \cdot \log m)$ where m is the distinct number of subjects. The number of triples is always much greater than the number of distinct subjects, we can safely ignore it.

In algorithm 2 after getting the most important subjects, we allocate the triples corresponding to that subject to a cluster. We then obtain k fragments. To allocate the remaining triples to these fragments, we need to find out the degree of closeness of each triple with the existing fragments. Given a triple, t not yet assigned to any fragment, we find out which fragment has the most number of triples which contain the object equivalent to the subject of triple, t . The triple t and all other triples which share the same subject which we call the secondary master node are then added to that cluster. This method is continued for the rest of the triples. Here each of the triple not part of initial partitioning is compared with every existing triple of the partitions. We do the preprocessing of partitions and use HashMap to record the occurrences of objects in a partition. Since every subject of the remaining triple is checked in the HashMap, the worst time complexity of the algorithm becomes $O(n \cdot k)$ where n is the number of triples and k is the number of fragments or partitions.

Algorithm 2: Semantic Aware Partitioning**Input:** RDF Triples, frequent subjects, k fragments**Output:** k partitions

```

1 Function SAPartition(RDF Triples, frequent subjects, k
   fragments):
2   for each fragment f=1 to k do
3     | fragment fi = matches of triples with subject si
4   make hashmap h for each fragment i where key = object
   and value = frequency for rest of the triples do
5     | put triple in fragment with maxhi(object)
6   return k fragments

```

3.2 Distribution using Partial Replication

When the user submits a query to the coordinator node, it will be answered using graph traversal from all the available nodes in the distributed environment in LBSD. This section includes details of the replication and distribution strategy.

After the fragmentation of the dataset, it is not necessary that we get fragments that are almost equal in size primarily because the frequency of outgoing edges is not uniformly distributed in the triples. While some nodes might have a high number of outgoing edges, others might barely have that many outgoing edges. This might lead to skewed distribution, which will result in unequal load distribution and delayed query execution time. To mitigate this problem, we calculate the sizes of the fragments and allocate them to different sites in such a way that there is an approximately equal load at each of the sites. So, a fragment of bigger size should be placed with a fragment of smaller size.

Algorithm 3: Distribution of Clusters**Input:** fragments**Output:** clusters

```

1 Function Allocation(fragments):
2   Array A of fragment, size of fragment for every
   fragment f do do
3     A.insert(F, sizeof(F))
4   sort A in descending order of size for every index i in A
   do do
5     Add A[i].fragment to cluster with lowest size

```

For Replication, LBSD uses centrality measurement. Degree centrality measures the number of incoming and outgoing relationships from a node. The Degree Centrality algorithm can help us find popular patterns(subject-object) in a graph. This is built-in feature of Neo4j[16]. For each predicate, we can measure the centrality that how much they are connected to subjects. Having the same centrality predicates should lie on the same host in a distributed environment. Here for our experiment, centrality lies between interval [0.14,1]. The highest centrality is 1, when the count of a number of subjects and number of labeled edges become same. Centrality helps to replicate and distribute data among available nodes.

Algorithm 4: Replication**Input:** Triples**Output:** Replicated Data

```

1 for each cluster c=1 to n do
2   for each Predicate p do
3     if cen(p) > Max[Ap] // Max[Ap]= centrality
       of max. occurring predicate
4     then
5       for i=1 to n do
6         Pi = p
7         Data will be replicated on node for cluster
           c[i=1 to n]

```

Replication replicates the data to the available nodes in the distributed system. Partial replication only replicates a few amounts of data that satisfy the given threshold value or cut-off. Here we have frequent patterns and its centrality. According to top k subjects analysed from algorithm 1, will have top k patterns. That means properties associated with those subjects. These top k patterns help to decide the replication level. So, the centrality of the top pattern becomes the threshold value for partial replication which is known as *Max. [Ap]*. For example, some subject *k1* is there in list of k subjects having centrality 0.58, then patterns of centrality between 0.58 to 1 will be replicated. Here in LBSD it is 0.65 i.e. patterns' centrality between 0.65 to 1 were replicated counted as most frequent one.

4 EXPERIMENTAL DETAILS

The hardware setup consists of Intel® Core (TM) i3-2100 CPU@ 3.10GHz 3.10 GHz 8GB. The software setup consists of Neo4j Desktop 1.1.10 [16] and for visualization neo4j browser version 3.2.19 is

used. We have used NeoSemantics[17] to upload rdf supported data files. As a distributed database we have used DGraph v1.0.13 [6].

4.1 Benchmark Dataset and Queryset

Linked Observation Data [15] (LOD) benchmark dataset is used for the experiment. LOD has near about 3000 categories. From that, we have used Linked Sensor Data(LSD) comprises of results of sensor observation results of Hurricane in the US. The dataset includes different observations of Wind Direction, Wind Gust, Air temperature, Humidity, Precipitation, etc.

The benchmark LSD query set is used for the experiments. It consists of 12 queries which are classified into four types. Type 1 and Type 2 queries are linear and star queries respectively. Type 3 and Type 4 queries are Administrative or Range queries and Snowflake queries respectively. There are 3, 4, 3, and 2 queries of Type 1, 2, 3, and 4 respectively. Linear queries select some predicates from data and Star queries select specific subject/objects relevant to the given predicate. Administrative or Range queries are used to retrieve data using aggregation function or range function and Snowflake queries are a combination of both Type 1 and Type 2.

4.2 Evaluation Parameters

Performance of LBSD will be evaluated using the following quantitative and qualitative evaluation parameters:

4.2.1 Quantitative parameters. This section discusses quantitative parameters that measure the performance of LBSD in terms of some percentage or value.

Algorithm Execution time (AET) is the time taken by the execution of all three algorithms of LBSD.

Inter-Node Communication (INC) is measured in terms of how much communication cost is there to answer a query using different nodes.

Query Execution Time (QET) is the time taken by a query to complete execution.

Query Join (QJ) measures the number of join operations to execute a query.

4.2.2 Qualitative parameters. This section discusses qualitative parameters which compare the LBSD in terms of quality measures.

Partitioning technique defines the technique used for the partitioning of data.

Distribution technique defines the technique used for the distribution of the RDF graph.

Workload information informs that is there any query workload information required for the execution.

Replication strategy defines the technique to replicate partitioned data.

Scalability defines how the system reacts when the data size increases. *Storage Requirement* gives an idea about amount of storage space used by system.

5 RESULTS AND DISCUSSION

LBSD is demonstrated using LSD benchmark data and query set. This section presents results for basic and scaled query execution

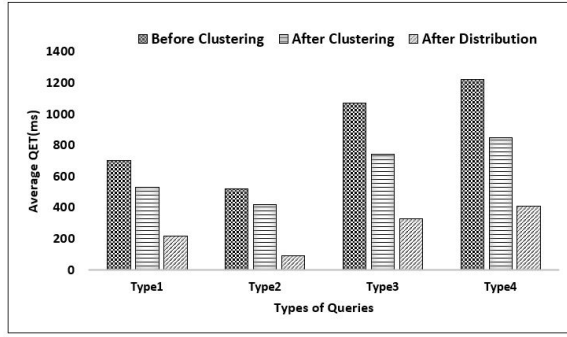


Figure 2: Average QET for all types

time, Algorithm execution time. It also contains discussions about the choice of replication level. The results for other quantitative parameters like query joins and INC are also included here.

For all the experiments of which results are presented in this section :

- Results are averaged over three consecutive executions to reduce fluctuations for each query.
- Data size is increased from 20K till 100K with a step size of 20K.

5.1 Basic Query Execution Time (QET)

QET analysis for LBSD has been done for LSD. There are four types of queries and results are taken by analyzing performance for each of them. QETs are averaged over three consecutive executions to reduce fluctuations for each query. Further all the QETs are averaged over all the queries of that type.

Figure 2 shows that Type 2 queries are taking less amount of time because it is just fetching the values whereas Type 4 queries are taking a larger amount of time compared to all types of queries as type 4 queries are snowflake queries which are complex compared to others.

5.1.1 Data Scaling for QET. Data scaling experiment done for the size 20k to 100k(1x to 5x). Figure 3 shows that QET increases with increase in the datasize from 20k till 100k for all the query types. This increase is more pronounced for Type 2 and Type 3 queries.

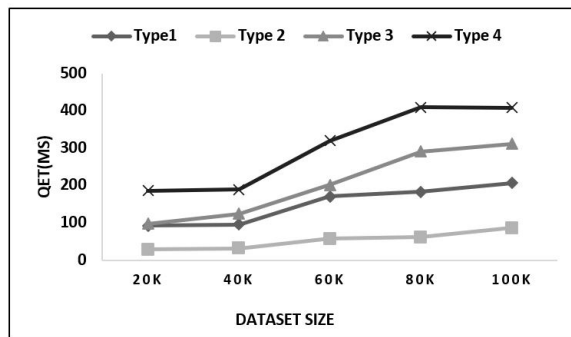


Figure 3: QET Before Partitioning

Type 2 queries taking a large amount of time when data size increases 40k to 60k as the value required to fetch is distributed over nodes. For all types of queries as data size increases QET is getting reduced after distribution as shown in Figure 4. There is on an average 71% of QET gain for all types of queries.

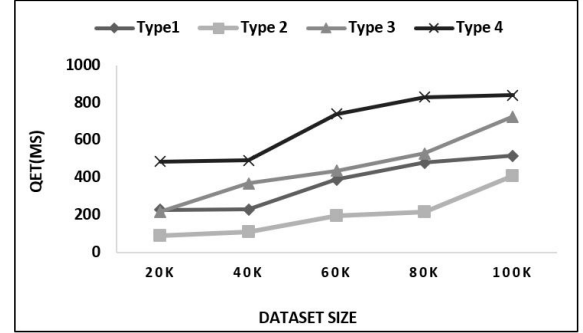


Figure 4: QET After Partitioning

5.2 Algorithm Execution Time (AET)

There are three algorithms used by the LBSD system. Algorithm 1 and Algorithm 2 are used in first phase and second phase of LBSD uses Algorithm 3 and Algorithm 4. The total execution time taken by the system to execute all four algorithms for different data sizes is shown in Figure 5. We can see that as data size increases AET increases. There is a ramp shown in the graph when data size increases from 80k to 100k(4x to 5x).

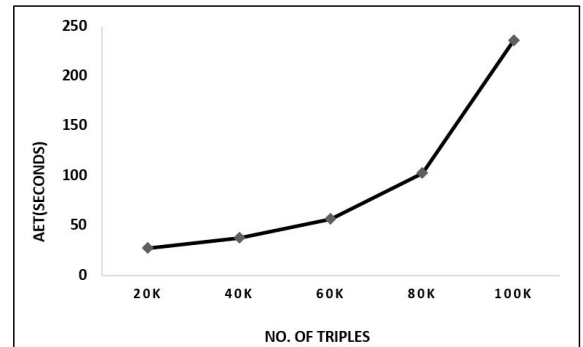


Figure 5: Algorithm Execution Time

5.3 Replication level

For partial replication, to decide replication level first we have kept threshold at centrality 0.65. As shown in Figure 6 there is a linear increment in no. of triples to be replicated with increasing data size. On average 12% of data were replicated. When we have changed the threshold value to 0.51, no. of triples increased with an average of 14% data were replicated. But for this experiment, It has been found that centrality 0.65 is optimal.

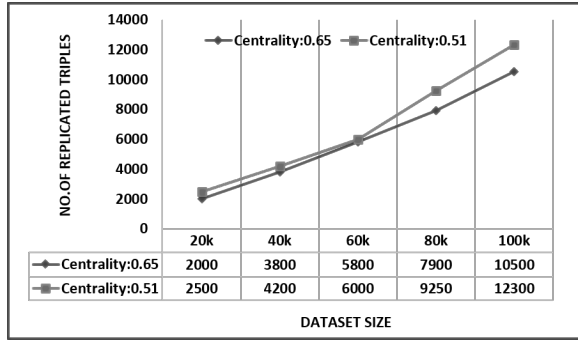


Figure 6: Replication level for centrality 0.65 and 0.51

5.4 Query Joins

If we compare LBSD to DWAHP[19] or to any such relational system, it works better in terms of Query Joins (QJ). In the graph database, we can access the whole database by traversing an edge, which reflects the absence of QJ. This is an advantage of LBSD that it eliminates QJ for accelerating queries over distributed data.

5.5 Inter Node Communication

Inter Node Communication (INC) means the amount of communication requires between available nodes in a distributed environment.

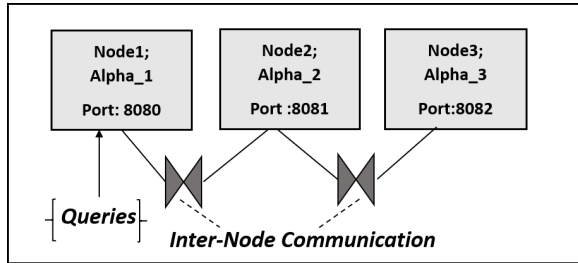


Figure 7: Inter Node Communication

As shown in Figure 7, INC is defined by communication cost that query uses to answer using three available nodes, named as; Alpha_1, Alpha_2, Alpha_3. These three nodes are working on a different port of DGraph Ratel. While accelerating queries over distributed data, out of 12 queries on an average of 7 queries are answered by the local node. So approximately 58% of queries are answered without INC.

5.6 Comparison of LBSD with the state of the art

LBSD is compared with different techniques including APR[2] which is graph based state of the art work and DWAHP[19], similar relational state of the art work. This section describes the detailed comparison of qualitative and quantitative parameters listed in Section 4.2.

The Table 1 shows the qualitative comparison of LBSD with APR and DWAHP. LBSD is a semantic aware partitioning technique whereas APR uses METIS [14], a simple weighted partitioning

technique. Semantic aware partitioning technique keeps semantic relation between nodes alive even after partitioning. Due to semantic relation, similar nodes will be on the same host which will accelerate linear and star queries. DWAHP is a relational approach which uses hybrid partitioning technique using a combination of property and binary tables. For the distribution, LBSD uses Centrality replication threshold whereas APR uses Global Query graph approach to identify border nodes communication cost. It helps to reduce communication cost between nodes on different hosts through replication.

The quantitative results for LBSD are shown in Table 2. The average QET gain for all types of Query reported 71%. The AET is averaged over all four algorithms and its total time complexity is $O(n)$. The INC is 58%, i.e. 58% queries are answered without INC. LBSD eliminated complex query join operations.

LBSD is a static partitioning approach and APR is an adaptive approach for distribution and partitioning. LBSD does not require workload information of the implementation whereas APR and DWAHP both requires workload information. For adhoc queries, workload aware system needs to re-run the algorithm for updated workload information. APR is implemented in such a way that it also takes space in consideration while distributing data. APR works well with storage adaption at three levels. It also uses compressed replication technique in which it compresses long URI to numerical value. DWAHP and LBSD use Partial replication technique. APR and DWAHP exhibit better scalability compared to LBSD.

QET for DWAHP and LBSD is almost the same. LBSD is reporting faster AET compared to DWAHP but it shows poor scalability beyond 4x. LBSD and APR being graph-based techniques, are able to eliminate Query joins. INC is approximately the same for both LBSD and DWAHP. APR generates small number of big clusters as compared to LBSD which reduces INC for APR. As a result of it, LBSD and DWAHP queries need to scan lesser data. APR replicates almost 30% of data whereas APR needs to replicate only 12% of data. As APR needs to rerun the algorithm for updated workload, the AET of LBSD reports faster than APR. There is an indexing overhead, as APR uses RDF-3X engine which requires indexing over all the three columns Subject, Object, and Predicate.

6 CONCLUSION

This method implemented to manage the increasing size of RDF data management by semantic aware partitioning and distribution of data using graph approach. Based on in-degree and out-degree of vertices LBSD partitions the data. For distribution purposes, we have distributed data on available three virtual nodes. LBSD compared in terms of two types of parameters: Qualitative and Quantitative. To analyze performance in terms of QET, the system uses 4 types of queries. It shows an average 71% gain for all types of queries after distribution. QET gain for type 2 queries in scalability experiments increases linearly with an average gain of 72% as it has lower INC whereas type 4 has an average gain of 55% as data size increases from 20k to 100k. The system also shows better performance in terms of inter-node communication as it answers 58% of the query by the local node. The scalability results show that AET increases rapidly when data size increases from 80k to 100k. Analysis of the rapid increment in AET is left for the future

Table 1: Qualitative Comparison of LBSD with state of the art

Parameters	LBSD	DWAHP [19]	APR [2]
Partitioning technique	Semantic Aware	Hybrid	Simple
Distribution technique	load balanced, Using replication Threshold	Not load balanced, Using n-hop reachability matrix	load balanced, Using Global Query Graph
Workload information	Not required	Required	Required
Replication technique	Partial replication	Partial Replication	Compressed replication
Scalability	Poor beyond 4x	Better	Better
Storage Requirement	Basic data + 12% replicated data	Basic Data + around 20% replicated data	Basic Data +30% replicated data RDF 3X indexing

Table 2: LBSD Results

Parameters	LBSD
QET	71%
AET	O(n)
INC	58%
QJ	eliminated

work. We can also make this system adaptive to deal with dynamic data which can also be considered as future work.

7 ACKNOWLEDGMENTS

We would like to thank Dr.Trupti Padiya, Postdoctoral Researcher, Friedrich Schiller University Jena for helping us to resolve technical issue during this research.

REFERENCES

- [1] D. J. Abadi, A. Marcus, S. Madden, and K. J. Hollenbach. Scalable semantic web data management using vertical partitioning. In C. Koch, J. Gehrke, M. N. Garofalakis, D. Srivastava, K. Aberer, A. Deshpande, D. Florescu, C. Y. Chan, V. Ganti, C. Kanne, W. Klas, and E. J. Neuhold, editors, *Proceedings of the 33rd International Conference on Very Large Data Bases, University of Vienna, Austria, September 23–27, 2007*, pages 411–422. ACM, 2007.
- [2] A. I. A. Al-Ghezi and L. Wiese. Adaptive workload-based partitioning and replication for RDF graphs. In S. Hartmann, H. Ma, A. Hameurlain, G. Pernul, and R. R. Wagner, editors, *Database and Expert Systems Applications - 29th International Conference, DEXA 2018, Regensburg, Germany, September 3–6, 2018, Proceedings, Part II*, volume 11030 of *Lecture Notes in Computer Science*, pages 250–258. Springer, 2018.
- [3] A. I. A. Al-Ghezi and L. Wiese. Uniadapt: universal adaption of replication and indexes in distributed RDF triples stores. In S. Groppe and L. Gruenwald, editors, *Proceedings of the International Workshop on Semantic Big Data, SBD@SIGMOD 2019, Amsterdam, The Netherlands, July 5, 2019*, pages 2:1–2:6. ACM, 2019.
- [4] Bio2rdf. <https://bio2rdf.org/>.
- [5] Dbpedia. <https://www.dbpedia.org/>.
- [6] Dgraph set up implemented using. <https://dgraph.io/>.
- [7] L. Galárraga, K. Hose, and R. Schenkel. Partout: A distributed engine for efficient rdf processing. In *Proceedings of the 23rd International Conference on World Wide Web, WWW '14 Companion*, page 267–268, New York, NY, USA, 2014. Association for Computing Machinery.
- [8] X. Guo, H. Gao, and Z. Zou. Wise: Workload-aware partitioning for rdf systems. *Big Data Research*, 22:100161, 2020.
- [9] N. L. Hoang, L. H. Trang, and T. K. Dang. A comparative study of the some methods used in constructing coresets for clustering large datasets. *SN Comput. Sci.*, 1(4):215, 2020.
- [10] K. Hose and R. Schenkel. Warp: Workload-aware replication and partitioning for rdf. In *2013 IEEE 29th International Conference on Data Engineering Workshops (ICDEW)*, pages 1–6, 2013.
- [11] A. Jain, T. Padiya, and M. Bhise. Log based method for faster iot queries. In *2017 IEEE Region 10 Symposium (TENSYP)*, pages 1–4, July 2017.
- [12] D. Janke, S. Staab, and M. Leinberger. Data placement strategies that speed-up distributed graph query processing. In *Proceedings of The International Workshop on Semantic Big Data, SBD '20*, New York, NY, USA, 2020. Association for Computing Machinery.
- [13] B. Kalavadia, T. Bhatia, T. Padiya, A. Pandat, and M. Bhise. Adaptive partitioning using partial replication for sensor data. In G. Fahringer, S. Gopinathan, and L. Parida, editors, *Distributed Computing and Internet Technology - 15th International Conference, ICDCIT 2019, Bhubaneswar, India, January 10–13, 2019, Proceedings*, volume 11319 of *Lecture Notes in Computer Science*, pages 260–269. Springer, 2019.
- [14] G. Karypis. *METIS and ParMETIS*, pages 1117–1124. Springer US, Boston, MA, 2011.
- [15] Linked observation data can be downloaded from . <http://wiki.knoesis.org/index.php/LinkedSensorData>.
- [16] Neo4j desktop downloaded from . <https://neo4j.com/>.
- [17] Neosemantics installation instruction available at: <https://github.com/neo4j-labs/neosemantics/blob/4.2/README.md>.
- [18] M. T. Özsu. A survey of rdf data management systems. *Front. Comput. Sci.*, 10(3):418–432, June 2016.
- [19] T. Padiya and M. Bhise. DWAHP: workload aware hybrid partitioning and distribution of RDF data. In B. C. Desai, J. Hong, and R. McClatchey, editors, *Proceedings of the 21st International Database Engineering & Applications Symposium, IDEAS 2017, Bristol, United Kingdom, July 12–14, 2017*, pages 235–241. ACM, 2017.
- [20] P. Peng, L. Zou, L. Chen, and D. Zhao. Adaptive distributed rdf graph fragmentation and allocation based on query workload. *IEEE Transactions on Knowledge and Data Engineering*, 31(4):670–685, 2019.
- [21] J. M. Schäfer, U. Sax, and L. Wiese. Benchmarking a distributed database design that supports patient cohort identification. In *Proceedings of the 24th Symposium on International Database Engineering and Applications, IDEAS '20*, New York, NY, USA, 2020. Association for Computing Machinery.
- [22] B. Shah, T. Padiya, and M. Bhise. Query execution for RDF data using structure indexed vertical partitioning. In *2015 IEEE International Parallel and Distributed Processing Symposium Workshop, IPDPS 2015, Hyderabad, India, May 25–29, 2015*, pages 575–584. IEEE Computer Society, 2015.
- [23] Q. Xu, X. Wang, J. Wang, Y. Yang, and Z. Feng. Semantic-aware partitioning on rdf graphs. In L. Chen, C. S. Jensen, C. Shahabi, X. Yang, and X. Lian, editors, *Web and Big Data*, pages 149–157, Cham, 2017. Springer International Publishing.
- [24] L. Zou and M. T. Özsu. Graph-based rdf data management. *Data Science and Engineering*, 2(1):56–70, Mar 2017.

Categorical Management of Multi-Model Data

Martin Svoboda
svoboda@ksi.mff.cuni.cz
Charles University

Faculty of Mathematics and Physics
Prague, Czech Republic

Pavel Čontoš
contos@ksi.mff.cuni.cz
Charles University

Faculty of Mathematics and Physics
Prague, Czech Republic

Irena Holubová
holubova@ksi.mff.cuni.cz
Charles University

Faculty of Mathematics and Physics
Prague, Czech Republic

ABSTRACT

In this vision paper, we introduce an idea of a framework that would enable us to model, represent, and manage multi-model data in a unified and abstract way. Its core idea exploits constructs provided by category theory, which is sufficiently general but still simple enough to cover any of the logical data models used in contemporary databases. Focusing on promising features and taking into account mature and verified principles, we overview the key parts of the framework and outline open questions and research directions that need to be further investigated. The ultimate objective is to pursue the idea of a self-tuning system that would permit us to collapse the traditionally understood conceptual and logical layers into just a single model allowing for unified handling of schemas, data instances, as well as queries.

CCS CONCEPTS

• Information systems → Entity relationship models; Semi-structured data; Data structures; Query languages.

KEYWORDS

Multi-model data, Category theory, Data modeling

ACM Reference Format:

Martin Svoboda, Pavel Čontoš, and Irena Holubová. 2021. Categorical Management of Multi-Model Data. In *25th International Database Engineering & Applications Symposium (IDEAS 2021), July 14–16, 2021, Montreal, QC, Canada*. ACM, New York, NY, USA, 7 pages. <https://doi.org/10.1145/3472163.3472166>

1 INTRODUCTION

The *variety* feature of Big Data inciting the so-called *multi-model data* has opened a challenging direction of data management. The (primarily) academia-driven approach, represented mainly by *poly-stores* [9], is based on the idea of *polyglot persistence*, i.e., the usage of a *mediator* managing a set of underlying database management systems (DBMSs), each being the best suitable candidate for a particular data model. On the other hand, there are (industry-driven) *multi-model DBMSs* [14] that offer the support of multiple models under the hood of a single system, treating all the data models as first-class

citizens [15]. Both the approaches have to face the same challenges brought by contradictory features of different but interlinked models and their data management specifics. For example, there are structured/semi-structured/unstructured formats, systems based on strong/eventual consistency, schema-full/schema-less/schema-mixed systems, declarative/functional query languages, etc.

In this vision paper, we describe the necessary steps that need to be carried out in order to provide a full-fledged and universally applicable solution to the problem of multi-model data management. We argue that a highly promising approach could be based on *category theory* [3], a theory general enough to cover all the currently popular data models (and probably even more) and having a sound mathematical background important for the efficient and correct data processing. Exploiting the constructs it offers, we provide a vision of a complex framework that would allow us to merge the conceptual and logical layers into just a single model through which schema descriptions, data instances, as well as query expressions could be handled in an entirely uniform and abstract way.

Based on the inspirational related work and with the help of a set of illustrating examples, the main contributions of the paper are: (1) discussion of ways how category theory can bring answers and solutions to the aspects of schema modeling, data representation, querying as well as evolution, transformations or migration, (2) identification of the related open questions and research directions that need to be further investigated so that the framework can be applied in polystore and multi-model scenarios, and (3) the actual description of the envisioned category-based unified autonomous DBMS, including its main advantages with respect to the contemporary systems.

Section 2 provides a brief introduction to category theory, Section 3 then thoroughly introduces the key parts of the framework, so that in Section 4, we summarize advantages of the category-based unified DBMS and conclude.

2 CATEGORIES

Formally, a category $C = (O, M, \circ)$ consists of a set of objects O serving as graph vertices, a set of morphisms M acting as directed edges, and a composition operation \circ for the morphisms.

Each morphism is modeled and depicted as an *arrow* $f : A \rightarrow B$, where $A, B \in O$, A being referenced to as a *domain* and B as a *codomain*, respectively. Whenever $f, g \in M$ are two morphisms $f : A \rightarrow B$ and $g : B \rightarrow C$, it must hold that $g \circ f \in M$, i.e., morphisms can be composed using the \circ operation and the composite $g \circ f$ must also be a morphism of the category (i.e., transitivity is required). Moreover, \circ must be associative, i.e., $h \circ (g \circ f) = (h \circ g) \circ f$ for any morphisms $f, g, h \in M$, $f : A \rightarrow B$, $g : B \rightarrow C$, and $h : C \rightarrow D$. Finally, for every object A , there must exist an *identity* morphism

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada
© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-8991-4/21/07...\$15.00
<https://doi.org/10.1145/3472163.3472166>

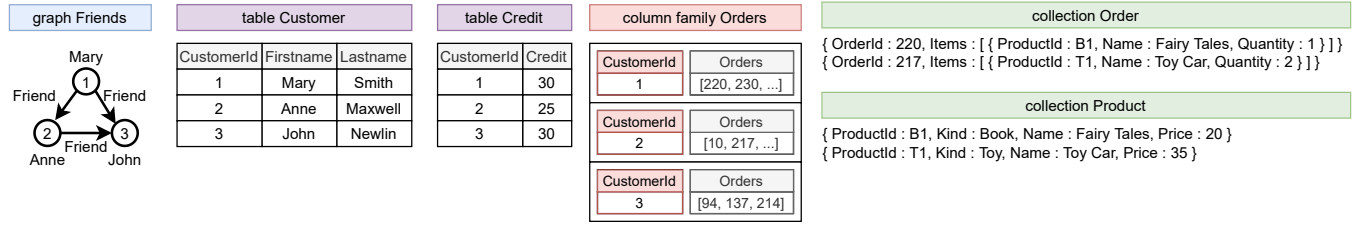


Figure 1: Sample multi-model scenario

1_A such that $f \circ 1_A = f = 1_B \circ f$ for any $f : A \rightarrow B$ (obviously serving as a unit with respect to the composition).

As an example, let us mention at least the **Set** category, where objects are arbitrary sets and morphisms functions between them. Note, however, that both objects and morphisms can, in general, represent abstract entities of any kind.

Considering the related work, category theory is a promising solution for the indicated issues. Existing *bottom-up* approaches start from a single logical model (relational [20, 21], object-relational [24], or CSV/document/RDF [23]) and define a respective schema category and operations using standard categorical approaches (such as functors). A *top-down* approach from [12] defines a schema category covering various conceptual modeling approaches, but only with respect to the most common model of that time – the relational model. In this paper, we also follow this direction, however, with respect to multiple interlinked data models at a time. In other words, we will show that the categorical approach can also be used for the multi-model world.

3 FRAMEWORK

The objective of the following text is to describe features, requirements, and principles of the core components of the envisioned database framework that would allow us to formally grasp the existing scenarios of polystores and multi-model databases, as well as could potentially contribute to the proposal of a truly unified and conceptual database system.

3.1 Multi-Model Scenario

In order to demonstrate the intended functionality, let us assume a sample multi-model scenario we will use throughout the individual illustrative examples.

EXAMPLE 1. Figure 1 provides an example of a multi-model scenario. The relational model (violet) contains general information about customers, whereas the graph model (blue) captures their mutual friendship. The document model (green) maintains orders which are bound with particular customers using the wide-column model (red). □

The range of the available and widely used logical models is, of course, wider than the particular models we incorporated into our sample scenario. While the traditional relational model was and still is used as the primary option, a recently emerged family of NoSQL systems enabled wider usage of key/value, wide column, document, and graph models, too. As we have mentioned, their contradictory features, unfortunately, increase the complexity of handling the multi-model data in a truly universal way.

For the purpose of describing schemas of data at the conceptual layer (which intentionally abstracts and conceals specific details of the underlying logical models), existing mature and frequently used modeling languages such as ER [5, 17] and UML [19] (class diagrams in particular) can be exploited (see Figure 2 for the sample scenario). Unfortunately, while ER is more expressive and suitable for describing complex real-world relationships, it is not well-formalized and exists in various notations differing not just visually. On the other hand, UML is standardized, but only too data-oriented (lacking, e.g., weak entity types or other constructs).

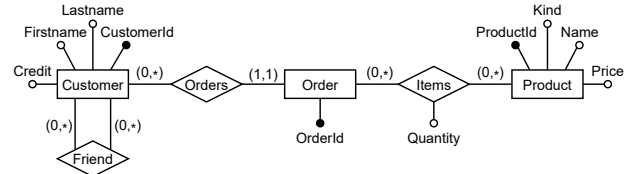


Figure 2: ER schema diagram for the sample data

Although this probably should not be the case, the main disadvantage of both the existing conceptual approaches is that they are actually not entirely suitable for grasping distinct data structures assumed by different logical models, simply because they are actually very closely related and inspired by the traditional relational model. This means they are built on top of the notion of ordinary sets of tuples in the first normal form, and so clearly not fully conforming to the principles and nature of the other models that permit, e.g., repeated values, union types, or hierarchically constructed properties.

There actually already exist papers dealing with ER modeling of multi-model data (e.g. [4, 10]), primarily for polystores. Unfortunately, they lack crucial details of mapping from the conceptual layer to the particular DBMSs, which strongly influences inter-model references, model overlapping, query optimization, evolution management, etc.

3.2 Schema Modeling

As a consequence of the previous observations, a new strategy capable of unified multi-model schema description and data representation needs to be found. We believe that an approach based on category theory could be promising enough, not just because of its universality and formal theoretical background, but also because of the already outlined existing approaches, though they are suffering from various drawbacks and are not yet ready for the multi-model scenarios.

The following idea of a *schema category*¹ can form the basis for the categorical conceptual modeling via which we will be able to describe the intended structure of the data together with basic integrity constraints. Borrowing the terminology from ER, objects of this category will represent individual entity types, attributes, as well as relationship types (without necessarily needing to distinguish between them later on, in the optimal case). In order not to disallow entity types with several or composed identifiers, super-identifiers can be exploited to encompass all of them, if needed. Morphisms will then interconnect the corresponding objects, allowing us to model the traditional concept of relationship cardinalities and attribute multiplicities through (min, max) constraints, where $min \in \{0, 1\}$ and $max \in \{1, *\}$. Obviously, $min \in \{0, 1\}$ would restrain the lower bound of a number of occurrences (optional vs. compulsory) and $max \in \{1, *\}$ the upper bound (at most one vs. at least one).

In order to make our category visualizations easier, we follow the convention that identity and non-core morphisms belonging to the transitive closure over the composition operation are entirely omitted, and that morphisms are labeled only with cardinalities different from $(1, 1)$, being treated as the default.

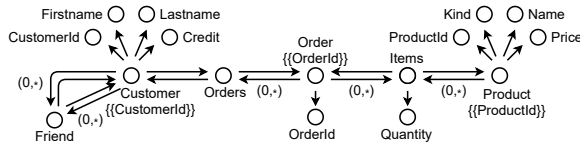


Figure 3: Schema category for the sample data

EXAMPLE 2. In Figure 3, we can see the schema category corresponding to the ER model from Figure 2. For instance, the object (node) *Customer* represents the entity type *Customer*. Neighboring objects connected via morphism (edges) correspond to its attributes (*Firstname*, *Lastname*, etc.) as well as relationship types (*Orders* and *Friend*) with respective cardinalities. □

Even though schema categories could also be designed directly by database users, this strategy might not be considered convenient enough, not just because of technical aspects that need to be carefully followed and ensured. Thus the outlined concept should primarily be understood as a means for internal database schema representation rather than a modeling language as such. This, in other words, means that a corresponding user-friendly modeling language would need to be proposed, too.

Moreover, in order to enable the option of deploying the envisioned framework in the context of the existing systems, transformations of ER and UML schemas, as well as at least semi-automatic inference methods capable of deriving schema categories from sample input data, would need to be proposed, too. More broadly, principles of generic schema management should be followed, similarly as in [1].

¹For the sake of simplicity, we omit technical details and more complex constructs. A separate paper [22] is devoted to the proposal of the schema category and the transformation process. In addition, the category itself can be designed variously, which is one of the open problems of the idea.

3.3 Database Decomposition

The very idea behind polystore and multi-model database scenarios is that different parts of the data are logically modeled, physically stored, and further processed differently by means of the corresponding logical models and query languages they are accompanied with. Having the schema category, the next step is to decompose it into such parts, let us call them *database components*. In practical terms, each of these components is expected to correspond to one of the underlying systems involved in a polystore (each one of them is accessible and queryable, yet specifically), and logical models involved in a multi-model database (with data directly and natively retrievable), respectively.

Each of these components would consist of a set of particular selected objects and morphisms, as if forming kind of a subgraph of the entire category that permits to incorporate the individual morphisms even without their ending objects (domain and codomain). Moreover, these components might be and most likely will often tend to intentionally be disconnected and/or more-or-less overlapping in real-world use cases. While the former aspect would require to take into account also the derived morphisms during the decomposition process, i.e., non-core morphisms derived using the category composition operation, the latter one finds its applicability in deliberate data redundancy across logical models, and so potentially improving query evaluation efficiency for data that is expected to be accessed together.

EXAMPLE 3. Sample schema category decomposition into particular data models (depicted using the colors from Figure 1) is visualized in Figure 4. For instance, the document (green) component covers orders. The graph (blue) and relational (violet) components partially overlap, since we maintain the respective attributes (*Firstname* and *CustomerId*) in both the models. □

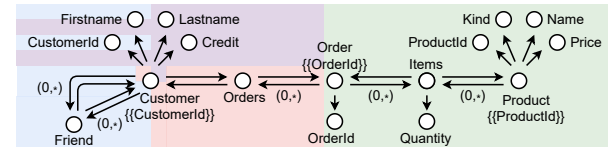


Figure 4: Decomposition into database components

Focusing on a particular component, its objects and morphisms will then need to be internally mapped to particular logical constructs and structures provided by a given model, e.g., tables (together with their columns, primary keys, foreign keys, or other constraints) in case of the relational model, or collections of JSON documents (and their objects, arrays, and values) in case of the document model assumed by MongoDB.

EXAMPLE 4. Let us consider the document (green) component in Figure 4. There exist several ways how to map this part of the category to one or more documents. Using a (semi-)automatic approach, the developer may choose, e.g., the following JSON schema (expressed symbolically):

```
{ OrderId, Items: [ { ProductId, Name, Quantity } * ] }
{ ProductId, Kind, Name, Price }
```

3.4 Data Representation

Data assigned to each of the outlined database components is stored and logically represented differently, thus we have to find a way

how such diversity could be encompassed within just one kind of a unifying data structure that would be capable of handling all the specifics of the individual models, but would still be simple enough. It is also worth realizing that this structure should serve not just as a means of modeling the data actually present in the database (and so conforming to its conceptual schema), but also a means to represent results of the evaluated query expressions, including the intermediate ones.

While the existing multi-model DBMSs [14] more or less painfully create an extension of the data structures used for the original single model, there exist proposals of more general approaches, too. E.g., the *NoSQL Abstract Model (NoAM)* [2] represents the data as named collections, each containing a set of blocks consisting of a non-empty set of entries. *Associative arrays* [8] are then defined as mappings from pairs of unique (column and row) keys to values. Or the *Tensor Data Model (TDM)* [11], which introduces the idea of generalized matrices. We believe, however, we can go significantly further.

It would only be beneficial if the desired data structure could be derived from the outlined schema category. Such an objective could be achievable through an *instance category*. The idea is that this category would have exactly the same objects and morphisms when compared to the corresponding schema category, they would only differ in what the objects and morphisms mean and contain, i.e., what they are supposed to internally model. Perhaps tables, i.e., sets of tuples, could be utilized as a basis, at least for now. Hence, the idea is that each entity/relationship/attribute object could contain a set of values belonging to the active domain, and morphisms mappings of pairs of such values.

EXAMPLE 5. Figure 5 provides a part of the instance category bearing the particular information about each product, i.e. ProductId and Name. \square

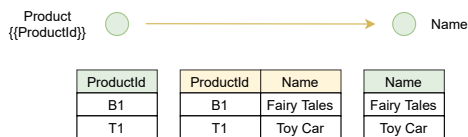


Figure 5: Part of an instance category

Obviously, it may not have been a good direction to stick with the principles of the relational model since complications at least with the ordering of values, duplicates, or non-trivial cardinalities of sub-attributes will unavoidably appear. Nonetheless, having the instance category proposed, a particular data format and techniques for data transformations will then also be needed (analogously to [13, 20]) so that a unified instance category can be obtained for input data in a particular format/model as well as such a category serialized in the opposite direction. Such low-level transformations will then find their essential position in data migration, evolution, or database self-tuning processes.

3.5 Query Language

The core functionality of each database system lies in its capability to query the data stored within it [26]. For obvious reasons, the

expressive power of the provided query language must be sufficient with respect to the intended purpose and usage [6], though it may significantly differ across the models, systems, and particular languages, too. Similarly, evaluation of query expressions as such must be efficient enough. The major limiting disadvantage of the existing settings is that users must be thoroughly aware of the logical schema of the data. It means that the languages and so the query expressions are, not surprisingly, tightly bound with the structure of the data to be queried. While this can be acceptable in single-model systems, it no longer is in truly multi-model ones.

Therefore we advocate for finding ways of querying that would genuinely be conceptual, though expressive enough from the practical point of view. In other words, the goal is to find a unified conceptual query language that would permit to query the data without any further knowledge of the database decomposition (possibly even changing through time) and regardless of the specifics of the individual involved logical models. This means one query language, one query constructs, one syntax. Moreover, it should also be closed with respect to the data model, i.e., that both extensional data as well as intermediate and final query results are modeled uniformly, via instance categories.

The envisioned querying could be based on sub-graph pattern matching widely exploited in graph databases, simply because graph models are broadly understood as the most complicated ones, as well as because categories in general are tightly related to graphs. The idea is to describe one graph pattern we are looking for in terms of a *pattern category*, consisting of only the selected objects and morphisms available in the corresponding schema category. The individual objects could be associated with selection conditions acting as domain filters. These conditions may be however complicated (with ranges, enumerations, conjunctions, disjunctions, ...), but may only involve a given object. Similarly, even morphisms could be equipped with conditions (comparisons, ...), either filtering or joining, always referencing only the pair of involved objects.

EXAMPLE 6. Figure 6 introduces a pattern category for a query that should return names, kinds, and prices of all books or toys which can be bought by a customer with name Mary. As we can see, the black parts correspond to the schema category objects and morphisms, and so the database contents, whereas the red part is a newly introduced morphism enabling to join the two fragments using a joining condition. Grey-filled circles represent projection (objects that should appear in the final result). \square



Figure 6: Pattern category for a sample query

Morphisms that were taken over from the schema category basically act as the corresponding inner joins, extensionally defined directly by the contents of the database. If it happens there are two or more disconnected parts (only assuming the adopted morphisms), they are joined using the Cartesian product (cross join), or a theta-join in case there are newly added morphisms with further joining conditions. It is also important to realize that particular schema objects and morphisms can be reused repeatedly in a query

pattern. All in all, everything described inside one pattern category is expected to be satisfied as if in conjunction, i.e., based on the all-or-nothing principle.

The notion of a pattern category obviously cannot be used in order to describe more complicated queries. Therefore the next natural step is to allow ourselves to construct a *query category* through which we would be able to combine several simple pattern categories by advanced query constructs and operations modeled as special query objects. That would hopefully permit to incorporate not just the following widely considered constructs: set operations over the patterns (union, intersection, difference), existential and general quantifiers including their negations, or optional patterns. Though there will undoubtedly be a variety of not just technical obstacles, we also need to be able to cover disjunctions of patterns through which complicated disjunctive conditions involving several objects could be expressed. Similarly, non-binary filtering or joining conditions, grouping and aggregation, or derivation of new values not present in the database via arithmetic expressions, function calls, etc.

Syntax of the actual query language as such will also need to be carefully designed so that it permits to express all the ideas but still remains user-friendly enough. While for the purpose of describing the individual pattern categories the idea of ASCII-art-inspired syntax assumed by Cypher query language in Neo4j could most likely be exploited, principles of the composition of more complex queries is a more significant challenge, possibly benefiting from the ideas of sub-queries or chaining of clauses, or simply the existing proposals such as SQL++ [16].

3.6 Query Evaluation

Based on the knowledge of the schema category, database decomposition, and internal schema mappings, the query category now needs to be decomposed into individual *query parts*, each to be evaluated separately so that it produces the corresponding intermediate result modeled in terms of instance categories with their schemas. Each query part consists of the selected objects and morphisms from the query category, yet not all of them do necessarily need to be involved. In fact, in the case of cross-model queries, evaluation of certain morphisms and/or query objects will intentionally need to be postponed, simply because no database component is capable of their evaluation on its own.

While in a polystore scenario each query part needs to be translated into the corresponding query expression within the specific query language a given system supports, and this expression internally evaluated so that the yielded result can be transformed into our unified logical representation, the corresponding intermediate result can directly be obtained from the database data files (at the physical layer) in case of a multi-model scenario.



Figure 7: Query category decomposition

EXAMPLE 7. Our decomposed inter-model query is depicted in Figure 7. The relational part (violet) extracts credits of customers named Mary:

```
SELECT T2.Credit
FROM Customer AS T1 NATURAL JOIN Credit AS T2
WHERE T1.Firstname = "Mary";
```

Analogously, using the MongoDB query language, the document part (green) extracts all properties but ProductId of books or toys:

```
db.Product.find(
  { Kind: { $in: [ "Book", "Toy" ] } }, { ProductId: 0 }
);
```

Note that multiple query parts belonging to the same database component may be generated, because the expressive power of a given query language may not be sufficiently high. For example, we can expect separate query parts and so multiple *find* queries for each of the involved collections in the case of MongoDB. Similar issues can appear in key/value or wide column databases, too.

Last but not least, while the query translation process takes a query category part and rewrites it to a specific query expression, we also need to deal with the opposite direction so that the existing query expressions in various at least widely used languages can be taken and migrated to our categorical representation.

Having a given query part evaluated, the obtained intermediate result needs to be represented as an instance category with a structure conforming to a newly defined schema category. For example, having an intermediate result in a form of a table retrieved from a relational system or a collection of possibly projected documents from MongoDB, its corresponding schema category could be constructed by *contracting* the given query category part (its objects and morphisms) into a star with all the involved attribute objects (simple or with sub-attributes) collected around the central object representing the super-identifier of the entire obtained result. The central object is derived from the *backbone* objects and morphisms of the query part, i.e., as a result of more-or-less complicated joining process.

The schema contraction step is essential in order to support advanced query constructs such as groupings, aggregations, or derivation of new values in general. As a consequence, however, we may lose the interpretability of certain objects or morphisms with respect to the original schema.

Once all the recognized query parts are evaluated and so the intermediate results obtained, the evaluation of all the postponed and not yet considered morphisms and/or query objects can be completed at the unified layer of the framework. This means that we need to evaluate these morphisms/objects one by one, partially contracting the corresponding parts of the current query category, step by step, so that we eventually retrieve the result of the entire query in the very end.

EXAMPLE 8. As depicted in Figure 8, the intermediate results for the relational (violet) and document (green) parts of the query were transformed into instance categories with the respective contracted schemas. By evaluating the postponed theta join, we retrieved the overall query result.

Evaluation and optimization of queries are no doubt absolutely crucial aspects defining the actual practical usability of database systems. Thus multiple heuristically designed query evaluation plans need to be considered so that the one with the lowest cost can be selected and actually executed as the only one. Apparently,

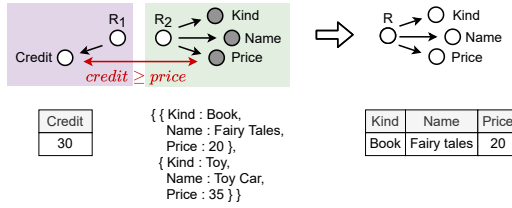


Figure 8: Intermediate and final query results

the efficient evaluation also needs to be supported by indices, in this case model-agnostics [15].

EXAMPLE 9. An alternative (though probably less efficient) query plan depicted in Figure 9 involves, in addition, also the graph model (blue), where names of customers can also be extracted. □



Figure 9: Alternative query evaluation plan

3.7 Evolution Management

Evolution management is a process of preserving the integrity of the whole system when user requirements change. A change in the structure of the data may require changes in related data instances, queries, integrity constraints, storage strategies, etc. Thanks to the general representation of all the data models using the schema category, we can define a unified cross-model set of schema modification operators (SMOs). The abstract categorical layer also enables backward compatibility of the queries even when the storage strategies and related mappings may change.

EXAMPLE 10. An example of evolution might involve merging of attributes Firstname and Lastname into one attribute Fullname. In addition, the data about customers can be migrated from the graph model to the relational model only. In Figure 10, we can see the old and new versions of the affected parts of the schema category. While the query accessing the modified part must be adapted accordingly in the first case, the schema category and so the categorical queries remain the same in the second case. □

The existing multi-model evolution management approaches provide an interesting inspiration, but they also have various significant limitations, such as, they only focus on aggregate-oriented NoSQL systems [7] or they omit critical inter-model links [18]. A sufficiently general set of SMOs can most likely be borrowed from [25].

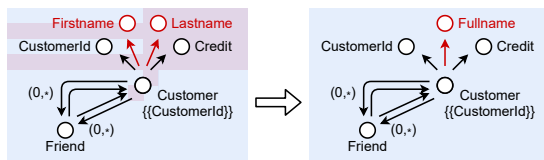


Figure 10: Old and new versions of schema category

4 SUMMARY AND CONCLUSION

The outlined framework has many features that seem to be promising and are undoubtedly worth further exploration. Either way, it can be deployed in both the polystore and multi-model scenarios, i.e., it is applicable to state-of-the-art systems. While we believe it can become a basis of newly designed database management systems, it still relies on, integrates, or extends various existing verified approaches, as well as it further elaborates concepts already advocating such objectives [15].

In other words, we propose to go beyond the ideas and boundaries of the contemporary systems toward entirely unified and conceptual databases that would allow us to fully abandon the idea of various logical models together with their distinct data structures, specific features, and often proprietary query languages, as it is demonstrated in Figure 11. Though such systems could be based on category theory, as we have outlined, other well-established formalisms could possibly serve as well.

It is apparent that there are only too many existing systems, models, formats, and languages. If it is difficult for the users to get sufficiently acquainted with such approaches and make decisions whether and when to use them in single-model situations, the more challenging the task it becomes when polyglot persistence is followed, i.e., when multiple models and systems should be considered, understood, deployed, and maintained at the same time. Obviously, at least from the long-term perspective, it is highly unlikely that such a variety together with currently ongoing trends could remain sufficiently sustainable.

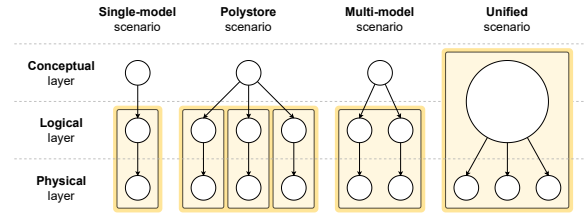


Figure 11: Layered architecture of database systems

Besides the already mentioned key aspects of the envisioned self-tunable database management system, we think the emphasis should be put on the formal background, unification, conceptuality, and user-friendliness. In other words, it is necessary to find a reasonable balance between the utilization of high-level formal theories such as the one assumed, and the practical applicability of the whole approach on the other hand. The core differences, features, advantages, and contributions represented by the framework we envisioned are summarized as follows.

- Intended schema of the data is described only once and at the conceptual platform-independent layer, only targeting real-world entities and relationships they can enter into.
- Schema categories have higher expressive power than the traditional ER or UML languages, seamlessly allowing to work with structured attributes or other constructs.
- Attributes, entity types, and relationship types no longer need to be mutually distinguished and handled differently, and so the entire modeling process can be simplified.

- Instance categories are general enough to serve as a universal data structure for all the currently widely used models, permitting the integration of future suitable models, too.
- The involved logical models play mutually equal roles, none of them is given priority, as is often the case in the existing multi-model systems.
- The traditional conceptual and logical layers collapsed into just a single unified representation, so they no longer need to be considered separately.
- Handling of the data variety aspect is pushed from the former logical layer to the physical one, where various file organizations and indices can appropriately be utilized.
- Decomposition into different models became internal with no impact on the users, and so they do not need to be aware of it, nor are they forced to adjust the style of thinking with respect to the particular models involved.
- Individual components can intentionally overlap each other, as well as do not necessarily need to contain all the data so that long-running migrations can be supported.
- Though the users can supervise the decomposition process, it can be fully autonomous and capable of reorganizing the data based on the changing workload or other aspects.
- Query language and its constructs are conceptual, and so independent on the internal representation of the data, not requiring the users to have its deeper knowledge.
- Querying itself is based on sub-graph pattern matching, a mature enough principle from graph databases allowing for the evaluation of complex queries.
- The query model is closed with respect to the input data and intermediate/overall results, all consistently modeled in terms of instance categories.
- Schemas, data, as well as queries are all modeled via homogeneously designed categories, all following the same structure and principles.
- Inter-model references or links of other kinds and cross-model queries are supported natively, which is not the case of many an existing system.
- Category theory provides a strong formal background, which brings the potential for advanced query optimization techniques that would otherwise be difficult to achieve.

Although we covered the key components concerning schema modeling, data representation, and querying, there are also other important aspects of multi-model data management we did not cover. We also did not want to provide a complex solution but to outline promising research directions and encourage both researchers and practitioners to pursue the envisioned ideas.

ACKNOWLEDGMENTS

The work was supported by the GAČR project number 20-22276S.

REFERENCES

- [1] Suad Alagić and Philip A. Bernstein. 2002. A Model Theory for Generic Schema Management. In *Database Programming Languages*. Springer, 228–246.
- [2] Paolo Atzeni, Francesca Bugiotti, Luca Cabibbo, and Riccardo Torlone. 2020. Data Modeling in the NoSQL World. *Computer Standards and Interfaces* 67 (2020), 103–149.
- [3] Michael Barr and Charles Wells. 1990. *Category Theory for Computing Science*. Vol. 49. Prentice Hall New York.
- [4] Francesco Basciani, Juri Di Rocco, Davide Di Ruscio, Alfonso Pierantonio, and Ludovico Iovino. 2020. TyphonML: A Modeling Environment to Develop Hybrid Polystores. In *MODELS '20* (Virtual Event, Canada). ACM, Article 2, 5 pages. <https://doi.org/10.1145/3417990.3421999>
- [5] P.P. Chen. 1976. The Entity-Relationship Model – Toward a Unified View of Data. *ACM Transactions on Database Systems* 1, 1 (March 1976), 9–36. <https://doi.org/10.1145/320434.320440>
- [6] Qingsong Guo, Jiaheng Lu, Chao Zhang, Calvin Sun, and Steven Yuan. 2020. Multi-Model Data Query Languages and Processing Paradigms. In *CIKM '20*. ACM, 3505–3506. <https://doi.org/10.1145/3340531.3412174>
- [7] Andrea Hillenbrand, Maksym Levchenko, Uta Störl, Stefanie Scherzinger, and Meike Klettke. 2019. MigCast: Putting a Price Tag on Data Model Evolution in NoSQL Data Stores. In *SIGMOD '19* (Amsterdam, Netherlands). ACM, 1925–1928. <https://doi.org/10.1145/3299869.3320223>
- [8] Jeremy Kepner, Julian Chaidez, Vijay Gadepally, and Hayden Jansen. 2015. Associative Arrays: Unified Mathematics for Spreadsheets, Databases, Matrices, and Graphs. *CoRR* abs/1501.05709 (2015). [arXiv:1501.05709](https://arxiv.org/abs/1501.05709)
- [9] Boyan Kolev, Raquel Pau, Oleksandra Levchenko, Patrick Valduriez, Ricardo Jiménez-Peris, and José Orlando Pereira. 2016. Benchmarking polystores: The CloudMdsSQL experience. In *BigData '16*. 2574–2579.
- [10] M. Kolonko and S. Müllenbach. 2020. Polyglot Persistence in Conceptual Modeling for Information Analysis. In *ACIT '20*. 590–594.
- [11] Eric Leclercq and Marinette Savonnet. 2019. TDM: A Tensor Data Model for Logical Data Independence in Polystore Systems. In *VLDB '18 Workshops*. Springer, 39–56. https://doi.org/10.1007/978-3-030-14177-6_4
- [12] Lippe, E. and Ter Hofstede, A. H. M. 1996. A Category Theory Approach to Conceptual Data Modeling. *RAIRO-Theor. Inf. Appl.* 30, 1 (1996), 31–79.
- [13] Zhen Hua Liu, Jiaheng Lu, Dieter Gawlick, Heli Helskyaho, Gregory Pogossians, and Zhe Wu. 2019. Multi-model Database Management Systems - A Look Forward. In *VLDB '18 Workshops*. Springer, 16–29. https://doi.org/10.1007/978-3-030-14177-6_2
- [14] Jiaheng Lu and Irena Holubová. 2019. Multi-Model Databases: A New Journey to Handle the Variety of Data. *ACM Comput. Surv.* 52, 3, Article 55 (2019). <https://doi.org/10.1145/3323214>
- [15] Jiaheng Lu, Zhen Hua Liu, Pengfei Xu, and Chao Zhang. 2018. UDBMS: Road to Unification for Multi-model Data Management. In *ER '18 Workshops (LNCS, Vol. 11158)*. Springer, 285–294. https://doi.org/10.1007/978-3-030-01391-2_33
- [16] Kian Win Ong, Yannis Papakonstantinou, and Romain Vernoux. 2014. The SQL++ Semi-structured Data Model and Query Language: A Capabilities Survey of SQL-on-Hadoop, NoSQL and NewSQL Databases. *CoRR* abs/1405.3631 (2014).
- [17] Atzeni Paolo, Stefano Ceri, Stefano Paraboschi, and Riccardo Torlone. 1999. Database Systems: Concepts, Languages and Architectures.
- [18] Marek Polák, Martin Nečáský, and Irena Holubová. 2013. DaemonX: Design, Adaptation, Evolution, and Management of Native XML (and More Other) Formats. In *IWAS '13* (Vienna, Austria). ACM, 484–493.
- [19] James Rumbaugh, Ivar Jacobson, and Grady Booch. 2004. *Unified modeling language reference manual*. Pearson Higher Education.
- [20] Patrick Schultz, David I. Spivak, Christina Vasilakopoulou, and Ryan Wisnesky. 2017. Algebraic Databases. *Theory & Applications of Categories* 32, 16-19 (2017), 547 – 619.
- [21] David I Spivak and Ryan Wisnesky. 2015. Relational Foundations for Functorial Data Migration. In *DBPL '15*. ACM, 21–28.
- [22] Martin Svoboda, Pavel Contos, and Irena Holubova. 2021. Categorical Modeling of Multi-Model Data: One Model to Rule Them All. In *MEDI '21: Proceedings of the 10th International Conference on Model and Data Engineering (LNCS, Vol. 12732)*. Springer, 1–8. https://doi.org/10.1007/978-3-030-78428-7_15
- [23] Laurent Thiry, Heng Zhao, and Michel Hassenforder. 2018. Categories for (Big) Data models and optimization. *Journal of Big Data* 5, 1 (2018), 1–20. <https://doi.org/10.1186/s40537-018-0132-9>
- [24] Chris Tuijn and Marc Gyssens. 1996. "CGOOD, a Categorical Graph-oriented Object Data Model". *Theoretical Computer Science* 160, 1-2 (1996), 217–239.
- [25] Michal Vavrek, Irena Holubová, and Stefanie Scherzinger. 2019. MM-evolver: A Multi-model Evolution Management Tool. In *EDBT '19*. OpenProceedings.org, 586–589.
- [26] Chao Zhang, Jiaheng Lu, Pengfei Xu, and Yuxing Chen. 2018. UniBench: A Benchmark for Multi-model Database Management Systems. In *TPCTC '18 (LNCS, Vol. 11135)*. Springer, 7–23. https://doi.org/10.1007/978-3-030-11404-6_2

Exploratory analysis of methods for automated classification of clinical diagnoses in Veterinary Medicine

Oscar Tamburis

Department of Veterinary
Medicine and Animal
Productions
University of Naples Federico II
Naples, Italy
oscar.tamburis@unina.it

Elio Masciari

Department of Computer Science
and Electrical Engineering
University of Naples Federico II
Naples, Italy
Institute for High Performance
Computing and Networks (ICAR)
National Research Council of Italy
Rende (CS), Italy
elio.masciari@unina.it

Gerardo Fatone

Department of Veterinary
Medicine and Animal
Productions
University of Naples Federico II
Naples, Italy
gerardo.fatone@unina.it

ABSTRACT

The present work describes the analysis conducted on the diagnoses made during the general physical examinations in the decade 2010–2020, starting from the DB of the EMR previously implemented in the University Veterinary Teaching Hospital at Federico II University of Naples. A decision tree algorithm was implemented to work out a predictive model for an effective recognition of neoplastic diseases and zoonoses for cats and dogs from Campania Region. The results achievable by data mining techniques for what concerns computer aided disease diagnosis and exploration of risk factors and their relations to diseases, show the increasing importance of Veterinary Informatics within the wider field of Biomedical and Health Informatics, and in particular its capacity to point out the existing connections between humans, animals, and surrounding environment, according to the One (Digital) Health perspective specifics.

CCS CONCEPTS

- Big Data applications; Biomedical and Health Informatics; Data Warehousing and OLAP; Knowledge Acquisition, discovery & Management

KEYWORDS

Electronic Medical Record; Veterinary Informatics; Decision tree algorithm; Data Mining; One Digital Health.

ACM Reference format:

Oscar Tamburis, Elio Masciari and Gerardo Fatone. 2021. Exploratory analysis of methods for automated classification of clinical diagnosis in Veterinary Medicine. In *Proceedings of 25th International Database Engineering & Applications Symposium (IDEAS 2021)*, July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 8 pages. <https://doi.org/10.1145/3472163.3472165>

1 Introduction

Animals, be they categorized as pets, livestock, or wildlife, stand as essential element in the evolution of human race for countless reasons. In particular, animal healthcare-related aspects play a prominent role because of their strict connections with human health. The monitoring of both wildlife and syntropic species' health state can provide in fact valuable information about (i) the quality of the environment they live in, and that they share with humans, in terms of pollution level, as well as food safety and traceability management; (ii) the occurring of zoonotic phenomena (for instance, leptospirosis and the recent COVID-19 pandemic). Furthermore, many non-infectious diseases (e.g. diabetes, cancer, and renal failure) are similar in both animals and humans [1]. Consequently, the need for an effective tracking of veterinary information to facilitate integration of animal medical data to support Public Health, has become essential. As a matter of fact, under the epidemiological perspective the advantages of using animals as sentinels or comparative models of human diseases are well known, as animals – or better, animal sentinels – may be sensitive indicators of environmental hazards and provide an early warning system for public health interventions [2]. With specific reference to Campania Region, this kind of studies are of particular concern due to the widely known so-called “Terra dei Fuochi/Land of Fires” phenomenon (see e.g. [3,4]). An as important aspect relates then to the control of the zoonoses, i.e. those diseases that can be transmitted from the animals to the human beings via faeces, urine, saliva, or blood. It is the

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 978-1-4503-8991-4/21/07...\$15.00

<https://doi.org/10.1145/3472163.3472165>

case of e.g. intestinal parasites and ticks (that use the animal as a vector), or rabies (transmitted via the saliva). Such risks have to be carefully taken into account when it comes to the cohabitation between humans and (conventional as well as non-conventional) pets [5]. The implementation of integrated veterinary information management systems (VIMS) for the capture, storage, analysis and retrieval of data, provides the opportunity for the cumulative gathering of the knowledge, and the capability for its competent interpretation [6]. To this end, it becomes useful to resort to data mining computational methods for extracting knowledge also in the case of animal large databases. Among the most diffused data mining algorithms [7,8], decision tree provides a tree-based classification for developing a predictive model according to independent variables [9].

In this paper the main results will be shown from the analysis of the data extracted from PONGO software ©, i.e. the first EMR solution implemented in the University Veterinary Teaching Hospital (it.: OVUD, acronym for Ospedale Veterinario Universitario Didattico) of the “Federico II” University of Naples, Italy. The main goal was to establish, by means of decision tree algorithm, a predictive model for an effective recognition of neoplastic diseases and zoonoses using clinical data, according to clinical, para-clinical, and demographic attributes. The investigation on the quality of clinical data of OVUD’s patients is intended for helping, at least on a region-wide scenario, to find out the presence of specific connections between people’s health, animal health, and their surrounding environment, thus conveying the specific Public Health dimension into the greater One (Digital) Health scenario [10].

2 Materials and Methods

2.1 Subjects

The data extracted from PONGO sw in form of MS Access DB relate to the general physical examination (GPE), that is the first visit performed from the veterinarian when the animal arrives to the hospital. The database contains about 10360 rows (one row per animal access) which span over a period going from 2010 to mid-2020. The visits were mainly performed on pets, i.e. dogs ($n = 8925$; 86%) and cats ($n = 1181$; 11%). Horses occurred to be treated in the hospital as well ($n = 160$; 2%). Only for a small part ($n = 92$; 1%) the animals examined belonged to other species (ducks, donkeys, bovines, buffaloes, goats, lagomorphs, rodents, tortoises, and birds). Besides animal species and date of the visit, the main fields of the DB also related to age and sex of the animal, main health issue (HI) acknowledged during the GPE, type of feeding (e.g. commercial vs. homemade), and vaccination status information. Also considered in the study were the kind of environments the animal used to live in (e.g. in an apartment, or outdoors), and the Italian province it came

from. As for the latter point, the research was limited to the provinces of Campania Region, due to the marginal number of rows related to patients coming from other Italian regions. Table 1 reports the accesses to OVUD, based on the geographic provenance, for dogs, cats, and horses. Several OLAP operations [11–14] were performed on the mentioned dataset, in order to investigate the quality of clinical data of OVUD’s patients for the considered time period.

Given the situation, it was decided to focus the investigation only on dogs and cats.

Table 1: Distribution of dogs, cats, and horses that accessed the OVUD, according to the Italian provinces

Species	Province	#	%
Dog	Avellino	145	2%
	Benevento	71	1%
	Caserta	753	8%
	Napoli	6967	78%
	Salerno	444	5%
	Other Italian provinces	545	6%
Cat	Avellino	22	2%
	Benevento	9	1%
	Caserta	68	6%
	Napoli	975	83%
	Salerno	41	3%
	Other Italian provinces	66	6%
Horse	Avellino	3	2%
	Benevento	7	4%
	Caserta	14	9%
	Napoli	64	40%
	Salerno	49	31%

2.2 Accesses per animal sex

Four types of sex specifications have to be considered for animals: male (M), castrated male (MC), female (F), spayed female (FS). Figure 1 reports the accesses to the OVUD of dogs and cats, respectively, for the time period considered. The number of rows/visits for which it was not possible to retrieve the sex of the animal, were also reported. Only in one case, the animal (dog) was reported as not visited after the access in the hospital. It has to be pointed out that the lower number of accesses registered in 2016 in both cases, was due to a partial stop of the OVUD activities, as a structural collapse interested at the end of 2015 part of the University Department that hosts the hospital itself. The number of male dogs’ accesses is about twice as much the female accesses in almost all the years considered, with quite lower numbers for the neutered dogs. A different situation concerns cats, where the differences M/MC and F/FS tend to be proportionally shorter, sometimes in favour of the neutered exemplars.

2.3 Health Issues per year

It was possible to identify about 140 different diagnoses from the GPE for the period considered. For mere space reasons, it was decided for both dogs and cats to investigate, for each year, only the three most relevant health issues (HIs), as reported in Tables 2 and 3. In case of HIs featuring the same occurrences, they were all considered. The only exception is for cats' HIs in 2012, where the occurrences for HI #3 were equal to 1 for a very large set of issues, so it was decided not to report it in the table.

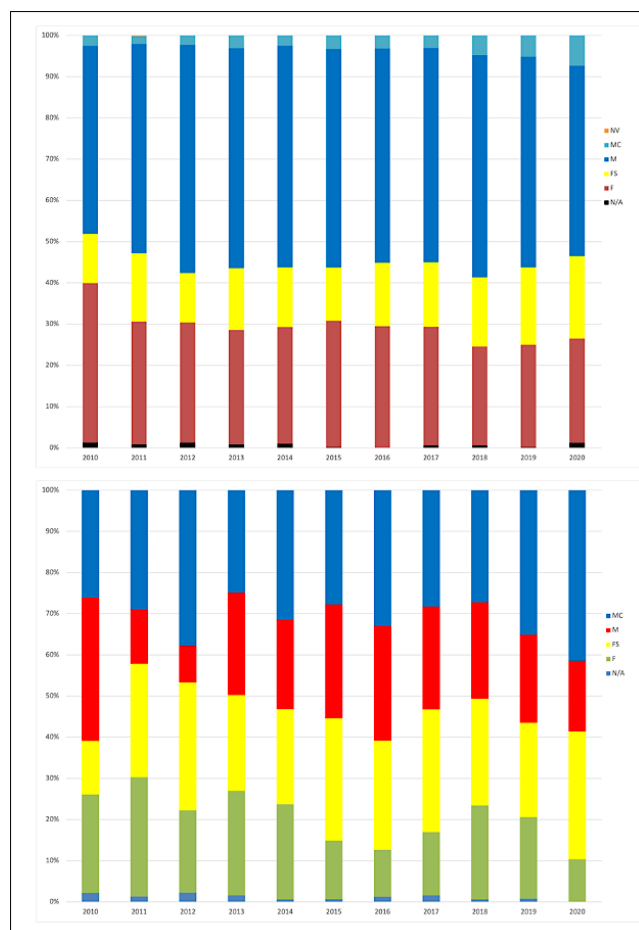


Figure 1: Accesses to OVUD of dogs (up) and cats (down)

Table 2: The three most diagnosed health issues for dogs

Year	HI #1	HI #2	HI #3
2010	Limping	Injury of abdomen	Firm lymph node (on exam.)
2011	Limping	On exam. - inspection of vomit	Pain in eye; Skin lesion (on exam.); Cough

Year	HI #1	HI #2	HI #3
2012	Alopecia	Skin lesion (on exam.); Limping	On exam. - inspection of vomit
2013	Limping	Alopecia	On exam. - inspection of vomit
2014	Limping	Neoplastic disease	Alopecia
2015	Limping	Neoplastic disease	Alopecia
2016	Limping	Injury of abdomen; Alopecia	Firm lymph node (on exam.)
2017	Limping	Firm lymph node (on exam.); Alopecia	Neoplastic disease
2018	Limping	Injury of abdomen	Neoplastic disease
2019	Injury of abdomen	Cough; Neoplastic disease	Limping
2020	Injury of abdomen	Alopecia	On exam. - inspection of vomit

Table 3: The three most diagnosed health issues for cats

Year	HI #1	HI #2	HI #3
2010	On exam. - inspection of vomit; Pain in eye	Injury of abdomen	Urinary tract pain
2011	On exam. - inspection of vomit	Alopecia	Urinary tract pain
2012	On exam. - inspection of vomit; Pain in eye	Firm lymph node (on exam.); Alopecia	.
2013	On exam. - inspection of vomit	Pain in eye; Injury of abdomen	Alopecia
2014	Pain in eye	Injury of abdomen	Urinary tract pain
2015	Alopecia; On exam. - inspection of vomit	Neoplastic disease	Skin lesion (on exam.); Pain in eye
2016	On exam. - inspection of vomit	Injury of abdomen	Alopecia
2017	Injury of abdomen	Skin lesion (on exam.)	On exam. - inspection of vomit; Urinary tract pain; Neoplastic disease

Year	HI #1	HI #2	HI #3
2018	On exam. - inspection of vomit	Alopecia; Pain in eye; Injury of abdomen	Skin lesion (on exam.)
2019	Injury of abdomen	Cough	Pain in eye; Alopecia
2020	Injury of abdomen	Firm lymph node (on exam.)	Skin lesion (on exam.); Closed fracture of hip; Sore mouth

The occurrences of such HIs during the years are reported in Tables 4 and 5, and in Figure 2.

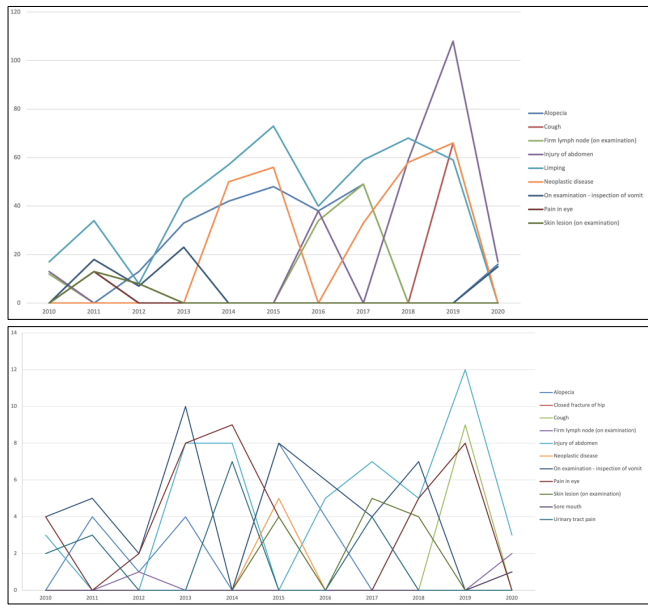


Figure 2: Distribution of the three most diagnosed health issues per year, for dogs (up) and cats (down)

Table 4: The three most diagnosed health issues for dogs

Year	Alopecia	Cough	Firm lymph node (on examination)	Injury of abdomen	Limping	Neoplastic disease	On examination - inspection of vomit	Pain in eye	Skin lesion (on examination)
2010	0	0	12	13	17	0	0	0	0
2011	0	13	0	0	34	0	18	13	13
2012	13	0	0	0	8	0	7	0	8
2013	33	0	0	0	43	0	23	0	0

Year	Alopecia	Cough	Firm lymph node (on examination)	Injury of abdomen	Limping	Neoplastic disease	On examination - inspection of vomit	Pain in eye	Skin lesion (on examination)
2014	42	0	0	0	57	50	0	0	0
2015	48	0	0	0	73	56	0	0	0
2016	38	0	34	38	40	0	0	0	0
2017	49	0	49	0	59	33	0	0	0
2018	0	0	0	59	68	58	0	0	0
2019	0	66	0	108	59	66	0	0	0
2020	16	0	0	17	0	0	15	0	0
TOT	239	79	95	235	458	263	63	13	21

Table 5: The three most diagnosed health issues for cats

Year	Alopecia	Closed fracture of hip	Cough	Firm lymph node (on examination)	Injury of abdomen	Neoplastic disease	On examination - inspection of vomit	Pain in eye	Skin lesion (on examination)	Sore mouth
2010	0	0	0	0	3	0	4	4	0	0
2011	4	0	0	0	0	0	5	0	0	0
2012	1	0	0	1	0	0	2	2	0	0
2013	4	0	0	0	8	0	10	8	0	0
2014	0	0	0	0	8	0	0	9	0	0
2015	8	0	0	0	0	5	8	4	4	0
2016	4	0	0	0	5	0	6	0	0	0
2017	0	0	0	0	7	4	4	0	5	0
2018	5	0	0	0	5	0	7	5	4	0
2019	8	0	9	12	3	0	0	8	0	0
2020	0	1	0	2	3	0	0	0	1	1
TOT	34	1	9	3	51	9	46	40	14	1

The total number of occurrences are depicted in Figure 3. In both cases, it is worth noticing the presence of neoplastic

diseases (dogs: N = 263; cats: N = 9) and firm lymph node-related (dogs: N = 95; cats: N = 3) diagnoses. Moreover, considering animals' age of birth (spanning from 1984 to 2020), it was possible to compare for each trimester the diagnoses of firm lymph nodes and neoplastic diseases. This revealed that the 44% cases of dogs of the same age, and the 5% cases of cats of the same age presented a number of occurrences of firm lymph node-related diagnoses greater or at least equal to neoplastic diseases diagnoses, thus inducing – at least for dogs – the reasonable hypothesis of an existing connection between the two pathologies.

Furthermore, Figure 3 reports the occurrences of those diagnoses which can be somehow related to the transmission of zoonoses, from tetanus (N = 1 for dogs) to vomit (dogs: N = 254; cats: N = 53). The number of occurrences of such diagnoses is the 7% of the total occurrences registered in OVUD for dogs and cats for the period considered.

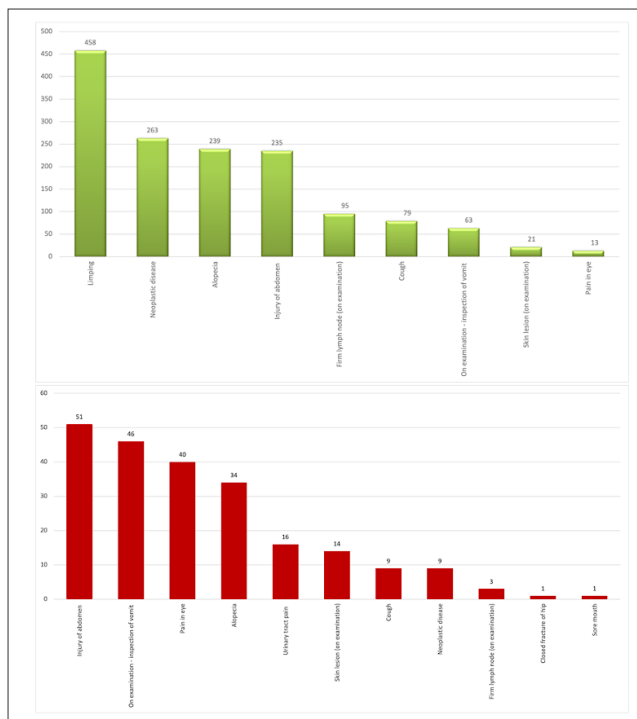


Figure 3: Total occurrences for the three most diagnosed health issues, for dogs (up) and cats (down)

2.4 Dataset

A preliminary step of dataset cleansing was necessary, especially for what concerns the health diagnoses, as no form of clinical standardized terminology had been deployed. Moreover, for about 30% rows (N = 3729), such type of data was missing, and only in a limited number of cases it was possible to get to it anyway by means of the analysis of the remainder fields of the database. Eventually, the total number

of participants considered in the model were 10108. Health Issues (as already done for reported in Tables 1 and 2) were categorized according to the vet-SNOMED terminology [15,16], as developed by the Veterinary Medical Informatics Laboratory at Virginia-Maryland College of Veterinary Medicine: for each of them, the corresponding Concept was identified, together with the related SNOMED hierarchy level, the Concept ID, and the Preferred Description (Synonym) ID [17,18]. Given the mentioned importance of identifying the presence of neoplastic diseases-related and/or zoonoses-related diagnoses, the need emerged to figure out a way to predict the presence of symptoms for both the issues considered – for both dogs and cats, who also happen to live very close to humans. In particular, according to what depicted in Figure 4, for what concerns zoonoses it was decided to consider for the analysis the diagnosis of “inspection of vomit”.

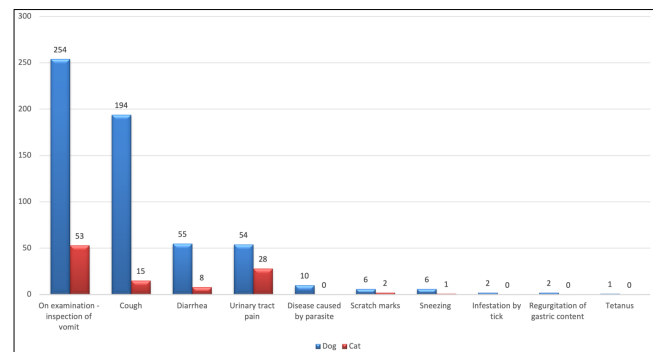


Figure 4: Total occurrences of zoonosis-related diagnoses, for both dogs and cats

Table 6 reports the vet-SNOMED classification for the two health issues considered in the study.

Table 6: Veterinary SNOMED classification of the investigated diseases

Type of Disease	Neoplastic Disease	On examination - inspection of vomit
Classification (Is a)	Disorder	Clinical finding
ConceptID (according to VTSL)	55342001	308637007
Preferred DescriptionID (according to VTSL and AAHA)	92007014	451951011
Acceptable synonym	Neoplasia	O/E - inspection of vomit

Type of Disease	Neoplastic Disease	On examination - inspection of vomit
Synonym's DescriptionID	1231453017	2670302013

2.5 DT ID3 Feature selection algorithm

The implementation of a Decision Tree Algorithm (DT) appeared as the most suitable way to investigate the membership of the subjects to different categories (diagnosed with neoplastic disease, or not; diagnosed with vomit, or not), taking into account the values of specific attributes (predictor variables), which in our case were identified for both cases as: animal sex, diet, vaccination, feeding routines, and living environment (plus the eventual presence of diagnosis of firm lymph nodes, for neoplastic diseases). In order to achieve these goals, a filter-based strategy using DT ID3 (Iterative Dichotomiser 3) was proposed [19]. As it is common in data mining methods to divide the dataset into two parts, also in our case the original sample was split into a training set (to train the model), and a test set (to evaluate the performance of DT ID3). In particular, the original Training dataset for the DT (oTrDS) featured all the accesses of dogs and cats to the OVUD between 2010 and 2018 ($N = 8643$; 86%), while the original Testing dataset (oTeDS) comprised the remaining accesses between 2019 and 2020 ($N = 1465$; 14%). The reason why it was not respected the common rule according to which oTrDS $\approx 70\%$ sampling data, and oTeDS \approx remaining 30%, mainly depends on two factors: (i) the reduced accesses to OVUD in 2016 due to the mentioned structure collapse, and; (ii) available data from year 2020 only cover the first six months. Since the aim of the study was to make prediction for two kind of health issues, each per two animal species, four specific Training datasets (sTrDS) and four specific Testing datasets (sTeDS) were extracted from oTrDS and oTeDS, respectively. For each case, a confusion matrix was used to evaluate the performance of the DT for classification of participants. Accuracy, sensitivity, and specificity were then measured for comparison. For sake of simplification, decision tree and confusion matrix have been represented in the following for one case only (presence of symptoms for neoplastic disease in dogs). A comparison was instead conducted for the performances of all four algorithms.

3 Results

A decision tree was built starting from the sTrDS related to the recognition of neoplastic disease for dogs ($N = 8927$). The sTeDS ($N = 1305$) was used to evaluate the model. The input variables were animal sex, diet, vaccination, feeding routines, living environment, and eventual presence of diagnosis of firm lymph nodes. As seen, since for dogs the possibility of a correlation was recognized between the diagnoses of neoplastic disease and firm lymph nodes, the number of subjects positive for both health issues (ND+ and L+) was

reported in the algorithm. ID3 uses two metrics to measure the importance of the input variables, or features, such as entropy (the measure of the amount of uncertainty) and information gain (the difference between the entropy of the DS, and the one related to the single feature). So, be DS a given dataset, and X the set of variables in DS. For each $x \in X$, the less the entropy, the more the information gain. For each iteration, the algorithm selects the feature with the smallest entropy/largest information gain value. The final decision tree with size 15, 8 leaves and 5 layers is shown in Figure 5.

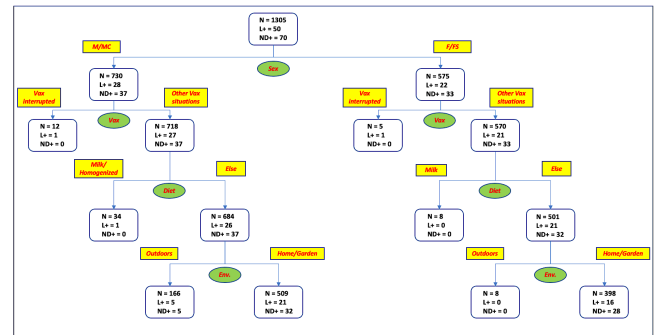


Figure 5: Decision Tree to evaluate the presence of symptoms for neoplastic disease in dogs

The evaluation of the tree was undertaken using confusion matrix on a testing dataset, as shown in Table 7. The algorithm had an Accuracy of 99%: of the 70 animals diagnosed as ND+ in the sTeDS, 60 were correctly classified using the DT. In a subordinate position, of the 50 animals diagnosed as L+, 37 were correctly identified. The specificity and sensitivity of the tree were equal to 99,2 and 1, respectively.

Table 7: Confusion Matrix of sTeDS related to the recognition of neoplastic disease for dogs

		Predicted Outcome	
		ND+	ND-
Expected Outcome	ND+	60 (TP)	10 (FP)
	ND-	0 (FN)	1235 (TN)

The performance of DT was also reported in Table 8. An overall comparison was instead conducted between the performances of the algorithm for the four cases investigated, as reported in Table 9.

Table 8: Performance of the DT ID3 model for the case investigated

Variable	Decision Tree Model
Sensitivity (95% CI)	1 (93,9 - 1)

Variable	Decision Tree Model
Specificity (95% CI)	99,2 (98,5 – 99,6)
Accuracy (95% CI)	99 (98,3 – 99,4)

Table 9: Comparison of the performances of the DT ID3 algorithm for all the cases investigated

		Accuracy	Sensitivity	Specificity
Dogs	Neoplastic Disease	99%	100%	99,2%
	Zoonosis (Vomit)	100%	100%	100%
Cats	Neoplastic Disease	100%	100%	100%
	Zoonosis (Vomit)	100%	100%	100%

Although the numbers of cats-related diagnoses extracted from the PONGO DB were significantly lesser than the dogs-related ones, the overall results obtained confirmed anyway the validity of the data mining algorithm implemented, which turned as highly capable of modelling the process of healthcare provision [20], as well as of setting forth reliable measurements of system performance and outcomes [21].

4 Discussion and Conclusions

In this paper a decision tree algorithm was implemented, starting from the database of the University Veterinary Hospital of the Federico II University of Naples, to work out a predictive model for an effective recognition of neoplastic diseases and zoonoses using clinical data, according to clinical, para-clinical, and demographic attributes. The main scope was to investigate whether and at what extent relations can stand between human and animal health, and their surrounding environments. The whole set of disciplines broadly dealing with the such kind of “connecting chain” goes under the name of One Health (OH), introduced for the first time as part of the twelve “Manhattan Principles” calling for an international, interdisciplinary approach to prevent diseases [22] and specifically animal-human transmissible and communicable ones. The raising interest towards the manifold competence areas of OH has led to the development of new disciplines such as digital epidemiology and public health informatics [23], with the purpose to improve understanding of health risks, effectiveness of management and policy decisions [24]. In this spirit, the seminal idea of “One Health Informatics” (OHI) was proposed as connected to the deployment of big data analytics to support and improve public health and medical research and address issues related to e.g. biodiversity control, disease monitoring, or control of zoonoses [25,26]. This was especially possible thanks to the

ongoing remarkable progress in, and the evolution of, the field of health and biomedical informatics (for humans), which is making increasingly manifest how health information technology (HIT) improves health, health care, public health, and biomedical research [27]. The same level of attention started increasing for veterinary disciplines as well, under the name of “veterinary (medical) informatics”: in the last years in fact, a significant development is being witnessed also for animal health for what concerns the information infrastructure that supports the knowledge translation processes of exchange, synthesis, dissemination, and application of the best clinical intervention research. In particular, the evolution of timely solutions of Electronic Medical Records allows achieving the same benefits as in human health, in terms of increased quality of care, better coordination among vet professionals, efficient care path design, etc. In particular, EMRs for pets (dogs and cats, mainly) share structure and objectives with human-focused products. Furthermore, new generations of vet-EMRs allow the recording of data as well for single animals from a herd, thanks to more functional interfaces with animals’ electronic identification tools.

Seen under this comprehensive point of view, the bursting of dynamics connected to the emerging and re-emerging of infectious diseases from national to supranational contexts, as well as the need to identify at global level risk factors and causes of health problems that arise at the human-animal-environment crossing, made even more remarkable the role of veterinarians towards the protection of human health. This points out therefore the growing of veterinary informatics, as also encompassing the need for new paradigms, approaches and technologies to reinforce the capacity of traditional surveillance systems for prevention and control of zoonoses, in terms of i.e. inter-sectoral coordination, link between human and animal health data and consequent management of flows of reliable data and information, or proper use of infrastructures, systems and human resources to detect outbreaks [28].

REFERENCES

- [1] Smith-Akin, K.A., Bearden, C.F., Pittenger, S.T., Bernstam, E.V., 2007. Toward a veterinary informatics research agenda: an analysis of the PubMed-indexed literature. *International journal of medical informatics*, 76(4), 306-312.
- [2] Vilhena, H., Figueira, A.C., Schmitt, F., Canadas, A., Chaves, R., Gama, A., Dias-Pereira, P., 2020. Canine and Feline Spontaneous Mammary Tumours as Models of Human Breast Cancer. In *Pets as Sentinels, Forecasters and Promoters of Human Health* (pp. 173-207). Springer, Cham.
- [3] Zaccaroni, A., Corteggio, A., Altamura, G., Silvi, M., Di Vaia, R., Formigaro, C., Borzacchiello, G., 2014. Elements levels in dogs from “triangle of death” and different areas of Campania region (Italy). *Chemosphere*, 108, 62-69.
- [4] Cavallo, S., Serpe, F. P., Rea, D., Pellicanò, R., D’Amore, M., Martinis, C. D., ... Baldi, L., 2018. The Land of Fires in Campania: the effects of exposure to dioxins on the progression of human breast cancer in an innovative animal model. In: XVIII Congresso Nazionale SI Di. LV, Società Italiana di Diagnostica di Laboratorio Veterinaria (SIDiLV), Perugia (PG), Italia, (pp. 41-42).

- [5] Mhlanga, A., 2020. Assessing the Impact of Optimal Health Education Programs on the Control of Zoonotic Diseases. *Computational and Mathematical Methods in Medicine*.
- [6] Plavšić, B., Nedić, D., Mićović, Z., Tešić, M., Stanojević, S., Ašanin, R., ... Milanović, S., 2009 Veterinary information management system (VIMS) in the process of notification and management of animal diseases. *Acta veterinaria*, 59(1), 99-108.
- [7] Ianni, M., Masciari, E., Mazzeo, G. M., Mezzananza, M., Zaniolo, C., 2020. Fast and effective Big Data exploration by clustering. *Future Generation Computer Systems*, 102, 84-94.
- [8] Masciari, E., 2012. SMART: stream monitoring enterprise activities by RFID tags. *Information Sciences*, 195, 25-44.
- [9] Bernardi, M. L., Cimitile, M., Martinelli, F., Mercaldo, F., 2017. A time series classification approach to game bot detection. In: *Proceedings of the 7th International Conference on Web Intelligence, Mining and Semantics* (pp. 1-11).
- [10] Benis, A., Tamburis, O., Chronaki, C., Moen, A., 2021. One Digital Health: a unified framework for future health ecosystems. *Journal of Medical Internet Research*, 23(2):e22189.
- [11] Pešić, S., Stanković, T., Janković, D., 2009. Benefits of using OLAP versus RDBMS for data analyses in health care information systems. *FACULTY OF ELECTRICAL ENGINEERING UNIVERSITY OF BANJA LUKA*, 56.
- [12] El Hajjami, S., Berrada, M., Harti, M., Diallo, G., 2020. Using Semantic Web Technologies and Multi-agent System for Multi-dimensional Analysis of Open Health Data. *Journal of Information & Knowledge Management*, 19(03), 2050021.
- [13] Masciari, E., 2009 (October). Trajectory clustering via effective partitioning. In *International Conference on Flexible Query Answering Systems* (pp. 358-370). Springer, Berlin, Heidelberg.
- [14] Manco, G., Masciari, E., Tagarelli, A., 2002 (November). A framework for adaptive mail classification. In *14th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2002)*. *Proceedings*. (pp. 387-392). IEEE.
- [15] Zimmerman, K.L., Wilcke, J.R., Robertson, J.L., Feldman, B.F., Kaur, T., Rees, L.R., Spackman, K.A., 2005. SNOMED representation of explanatory knowledge in veterinary clinical pathology. *Veterinary clinical pathology*, 34(1), 7-16.
- [16] Lerner, H., 2017. Conceptions of health and disease in plants and animals. *Handbook of the Philosophy of Medicine*. Dordrecht: Springer Science+ Business Media, 287-301.
- [17] SNOMED Homepage, www.snomed.org, last accessed 2021/06/14.
- [18] VTSL Home, <https://vtsl.vetmed.vt.edu>, last accessed 2021/06/15.
- [19] Tayefi, M., Tajfard, M., Saffar, S., Hanachi, P., Amirabadizadeh, A. R., Esmaeily, H., ... Ghayour-Mobarhan, M., 2017. hs-CRP is strongly associated with coronary heart disease (CHD): A data mining approach using decision tree algorithm. *Computer methods and programs in biomedicine*, 141, 105-109.
- [20] Tamburis, O., 2019. Bridging the gap between process mining and des modeling in the healthcare domain. In *2019 E-Health and Bioengineering Conference (EHB)* (pp. 1-4). IEEE, Iasi, Romania.
- [21] Luzi, D., Pecoraro, F., Tamburis, O., 2017. Appraising Healthcare Delivery Provision: A Framework to Model Business Processes. *Studies in health technology and informatics*, 235, 511-515.
- [22] Mackenzie, J.S., Jeggo, M., 2019. The One Health approach—Why is it so important? *Trop Med Infect Dis*. 4(2): 88.
- [23] Mavragani, A., 2020. "Infodemiology and Infoveillance: Scoping Review", *J Med Internet Res*, Vol.22, No.4, e16206.
- [24] VanderWaal, K., Morrison, R.B., Neuhauser, C., Vilalta, C., Perez, A.M., 2017. "Translating Big Data into Smart Data for Veterinary Epidemiology", *Front Vet Sci*, Vol.4, p. 110.
- [25] Ossebaard, H.C., 2014. One health informatics. In: *Proceedings of the 23rd International Conference on World Wide Web* (pp. 669-670).
- [26] Asokan, G.V., Asokan, V., 2015. Leveraging "big data" to enhance the effectiveness of "one health" in an era of health informatics. *Journal of epidemiology and global health*, 5(4), 311-314.
- [27] Mantas, J., Ammenwerth, E., Demiris, G., Hasman, A., Haux, R., Hersh, W., ... & Wright, G., 2011. Recommendations of the International Medical Informatics Association (IMIA) on education in biomedical and health informatics-first revision. *European Journal of Biomedical Informatics*, 7(2).
- [28] Choi, J., Cho, Y., Shim, E., Woo, H., 2016. Web-based infectious disease surveillance systems and public health perspectives: a systematic review. *BMC public health*, 16(1), 1-10.

Optimizing Transaction Schedules on Universal Quantum Computers via Code Generation for Grover's Search Algorithm

Sven Groppe

Institute of Information Systems (IFIS)
University of Lübeck
Lübeck
Germany
groppe@ifis.uni-luebeck.de

Jinghua Groppe

Institute of Information Systems (IFIS)
University of Lübeck
Lübeck
Germany
groppej@ifis.uni-luebeck.de

ABSTRACT

Quantum computers are known to be efficient for solving combinatorial problems like finding optimal schedules for processing transactions in parallel without blocking. We show how Grover's search algorithm for quantum computers can be applied for finding an optimal transaction schedule via generating code from the problem instance. We compare our approach with existing approaches for traditional computers and quantum annealers in terms of preprocessing, runtime, space and code length complexity. Furthermore, we show by experiments the expected number of optimal solutions of this problem as well as suboptimal ones. With the help of an estimator of the number of solutions, we further speed up our optimizer for optimal and suboptimal transaction schedules.

CCS CONCEPTS

• **Information systems** → **Data management systems**; **Database transaction processing**.

KEYWORDS

Quantum computing, transaction processing, synchronization, 2-phase-locking, database, schedule

ACM Reference Format:

Sven Groppe and Jinghua Groppe. 2021. Optimizing Transaction Schedules on Universal Quantum Computers via Code Generation for Grover's Search Algorithm. In *25th International Database Engineering & Applications Symposium (IDEAS 2021)*, July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 8 pages. <https://doi.org/10.1145/3472163.3472164>

1 INTRODUCTION

For the universal quantum computer, Grover [14] developed an algorithm for finding (with a high probability) the input to a black box function with a particular result to be searched for. In comparison to traditional computing, Grover's search algorithm achieves a quadratic speedup, which is to be optimal among all possible quantum algorithms [3, 14].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-8991-4/21/07...\$15.00
<https://doi.org/10.1145/3472163.3472164>

Grover's search often serves as basis for algorithms solving combinatorial optimization problems [2] by

- encoding candidates of the solution as an integer number, which is the input of the black box function of Grover's search, and
- using a function to check if a candidate solution has costs below a given threshold as black box function for Grover's search.

We analyze this pattern of solving combinatorial optimization problems on quantum computers with the optimizing transaction schedules problem [5, 6]. The optimizing transaction schedules problem is a variant of the job shop scheduling problems (JSSP), where jobs are assigned to machines (i.e., cores of a multi-core CPU in the domain of parallel computing jobs) with the optimization goal to minimize the overall processing time. For the optimizing transaction schedules problem, the jobs are transactions, which might block each other because of accessing at least one same data object, and conflicting transactions are scheduled to **not** run in parallel avoiding blocking of these transactions.

We propose to optimize the transaction schedule of batches by utilizing quantum computers as hardware accelerators [11–13]. By using pipelining we can optimize the transaction schedule already of the next batch during the processing of the current batch of transactions (see Figure 1). In this way, the quantum computer is run in parallel to the transaction processing on traditional CPUs in order to minimize waiting times and to maximize throughput.

Grover's search needs to call its black box function with candidate solutions in superposition. Only a limited set of operations can be applied to qubits in superposition, such that not all algorithms of traditional computing have corresponding counterparts in quantum computing. As a consequence for many combinatorial optimization problems and also for the optimizing transaction schedules problem, the code (i.e., the combination and orchestration of the quantum logic gates) of the black box function needs to be generated for the specific instance of the considered problem. We will show how to generate the code of the black box function for an instance of the optimizing transaction schedules problem.

Whenever the black box function doesn't need constant time, but its runtime depends on the instance of the considered problem, the overall runtime complexity of solving the considered combinatorial optimization problem is higher than the runtime complexity of the pure Grover's search. Hence, we will analyze the runtime complexity of solving the optimizing transaction schedules problem. Additionally, we analyze the preprocessing time (needed for generating the black box function), space requirements (i.e., number of qubits) and code length, and compare these results with the naive implementation on traditional computers (enumerating and testing

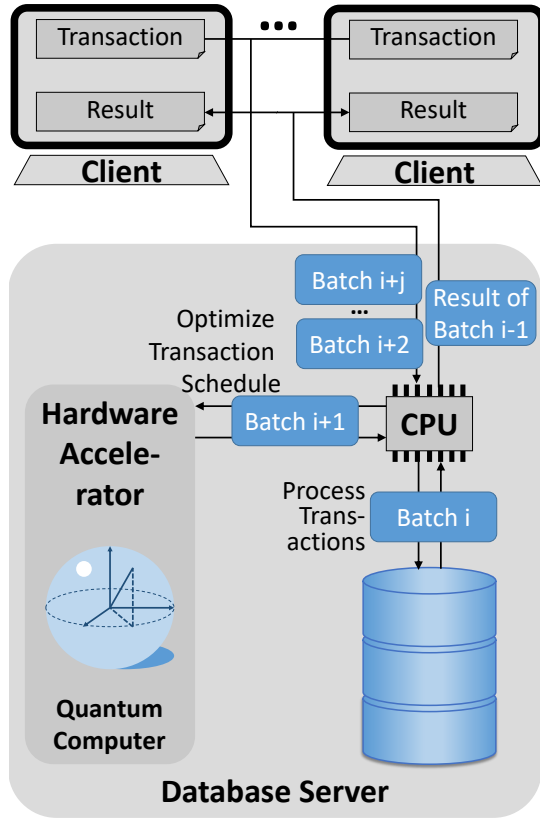


Figure 1: Pipelining the optimization of transaction schedules and the processing of transaction batches: The batches $i + 2$ to $i + j$ of transactions are in the queue for optimizing their schedule and processing, the schedule of batch $i + 1$ is currently optimized by the quantum computer, batch i is currently processed and the results of batch $i - 1$ are being transmitted to the clients.

all candidate solutions) and a recent solution [5, 6] on quantum annealers (specialized quantum computers for solving quadratic unconstrained binary optimization (QUBO) problems).

We have implemented an optimizer for transaction schedules by generating the code for the quantum simulator Silq [4] to show that the code generation is quite efficient and tests of the generated code are successful in Silq.

Our main contributions are

- a code generation approach for the black box function of Grover’s search running on universal quantum computers for optimizing transaction schedules,
- a complexity analysis concerning preprocessing and execution time, space and code length, and comparison of our approach with the ones running on traditional computers and on quantum annealers, and
- an implementation for the quantum simulator Silq [4].

2 BASICS

In this section, we introduce quantum computing in subsection 2.1, Grover’s search algorithm in subsection 2.2 and transaction management in subsection 2.3.

2.1 Quantum computer

Quantum computers are computers that are not based on classical mechanics, instead they exploit the effects of quantum mechanics.

Quantum mechanics describes the states and behavior of particles that are smaller than the size of an atom and do not follow the laws of classical physics. At this scale, there occur effects that the quantum computer makes use of, especially the principle of superposition and that of quantum entanglement. The quantum computer uses qubits, which can take on 2 states simultaneously due to the principle of superposition. While a bit can assume the state 0 or 1, a qubit assumes the states 0 and 1 simultaneously. If a measurement of the state is made, the qubit changes to one of the two states. Both states have relative probabilities with which they are assumed in the measurement. The principle of quantum entanglement enables the mutual influence of qubits, since entangled qubits mutually influence their probabilities. Imagining the principle of superposition as a special form of parallel computing opens up a new world of computation beyond polynomial time and allows in theory an exponential speedup compared to classical computers.

Universal Quantum computing finds desired solutions of a problem by clever manipulation of single qubits as well as entangled qubits. For the purpose of manipulating qubits, gates provide elementary operations on one or two qubits. For example, the Hadamard-Gate puts one qubit into superposition and a Controlled-NOT(CNOT)-Gate inverts a second qubit depending on the first [1]. A quantum computer is thus able to execute quantum algorithms like Shor’s algorithm for factorizing large numbers [19] or Grover’s algorithm for searching in huge unsorted databases [14, 15].

2.2 Grover’s search algorithm

Grover [14] showed that his search algorithm has an expected runtime of $\frac{\pi}{4} \cdot \sqrt{N}$ basic steps and claimed based on a result in [3] that up to a multiplicative constant among all possible quantum algorithms the optimality of his algorithm can be proven. Variants [7, 18] of Grover’s search propose applications running on average in $u \cdot \sqrt{\frac{N}{k}}$ basic steps without knowing the number k of solutions in advance. The variants differ upon being randomized [7] and deterministic [18], and the constant u , where [7] achieves $u = \frac{9}{4}$ and [18] $u = \frac{8 \cdot \pi}{3}$. Whenever the number k of solutions is approximately known in advance, then Grover’s search is able to find a solution in $\frac{\pi}{4} \sqrt{\frac{N}{k}}$ basic steps, such that a speedup of \sqrt{k} can be achieved in comparison to when only one solution is present, a speedup of 2.86 in comparison to [7] and 10.66 in comparison to [18].

2.3 Transaction Management

A **transaction** $t = \langle s_1, \dots, s_{|t|} \rangle$ of length $|t|$ is a series of operations s_i , carried out by a single user or application program, which reads or updates the contents of the database [9]. Each operation $s_i =$

$a_i(e_i)$ consists of the type of access $a_i \in \{r, w\}$, where r represents a read access and w a write access, and the object e_i to be accessed.

For example, the transaction $t = \langle r(A), w(A), r(B), w(B) \rangle$ is of length $|t| = 4$.

Since transactions often need to access a database at the same time and thereby often reference the same objects, transactions in a database system are required to fulfill the so-called ACID properties [9]. The focus here is on the fulfillment of the (I)solation property, which guarantees to avoid problems of unsynchronized parallel execution of several transactions and requires concurrently executed transactions to not influence each other.

2.3.1 Conflict Management. To ensure the isolation property, conflicts between transactions must be dealt with. The simplest strategy to deal with conflicts would be to execute the transactions serially. Since this is too slow for operational systems and also many transactions do not conflict with each other at all, in practice transactions are executed in parallel. There are various approaches to dealing with the conflicts that arise, one of them is the use of locks. For this purpose, each transaction acquires a lock for an object before access and releases it after access, thus preventing concurrent access to an object.

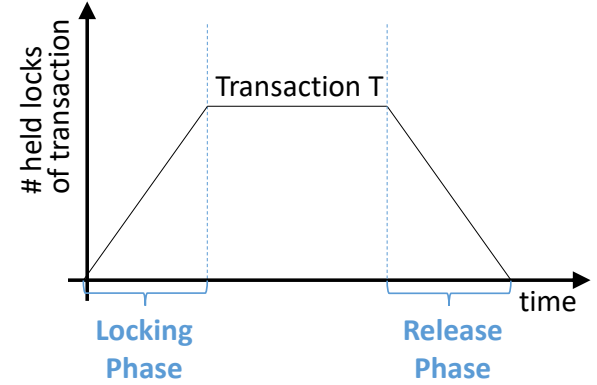
The **two-phase-locking protocol** (see Figure 2) is a locking protocol that requires each transaction consisting of two subsequent phases, the locking phase and the release phase. During the locking phase, the transaction may acquire locks but not release them, whereas the release phase requires the release of previously required locks. If a transaction has released a lock, it may not acquire any new ones. There is a distinction between *read locks* (also called *shared locks*) and *write locks* (also called *exclusive locks*). Variants of the protocol include the conservative two-phase-locking protocol and the strict two-phase-locking protocol. The conservative variant (*preclaiming*) requires that all locks that are required during a transaction are acquired before the transaction is started.¹ The strict variant holds all locks until the end of a transaction, which avoids *cascading aborts* occurring in case of so called *dirty reads* of objects written by transactions, which are later aborted resulting in an abort of the current transaction as well. In this contribution, we consider the combination of the conservative and the strict two-phase-locking protocol in our transaction model, which results in a serial execution of all transactions that access the same objects.

2.3.2 Conflicts. Let T be the set of transactions, D be the set of data objects. Two transactions $i \in T$ and $j \in T$ are in conflict with each other if there exists two operations of these transactions being in conflict with each other. Two operations $a_i(e) \in i$ and $a_j(e) \in j$ of the transactions i and j are in conflict with each other if they access the same object $e \in D$ and at least one of the operations is writing the object:

$$\begin{aligned} & a_i(e) \text{ in conflict with } a_j'(e) \text{ if} \\ & \exists i, j \in T, e \in D, a_i(e) \in i, a_j'(e) \in j : \\ & i \neq j \wedge (a_i = w \vee a_j' = w) \end{aligned}$$

¹Please note that for those transactions, for which the required locks are not known before processing, the required locks can be determined by an additional phase before transaction processing. The contribution in [20] describes such an approach which can be also applied in our scenario.

a) 2 Phase Locking



b) Strict Conservative 2 Phase Locking

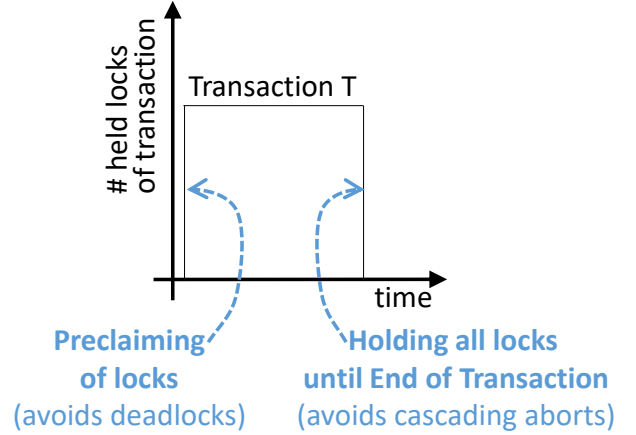


Figure 2: a) Two-phase-locking protocol with locking phase and release phase, and b) the strict conservative two-phase locking protocol

We assume that accesses to the same object $e \in D$ are serialized and we use the notation $a_i(e) \rightarrow a_j'(e)$ in order to denote that the operation $a_i(e)$ is executed before $a_j'(e)$. The isolation property is therefore fulfilled if all conflict operations of conflicting transactions $i \in T$ and $j \in T$ ($i \neq j$) are processed in the same order of transactions:

$$\begin{aligned} & \left(\forall a_i(e) \in i, a_j'(e) \in j : \right. \\ & (a_i = w \vee a_j' = w) \Rightarrow a_i(e) \rightarrow a_j'(e) \Big) \\ & \vee \\ & \left(\forall a_i(e) \in i, a_j'(e) \in j : \right. \\ & (a_i = w \vee a_j' = w) \Rightarrow a_j'(e) \rightarrow a_i(e) \Big) \end{aligned}$$

The two-phase-locking protocol fulfills the isolation property by guaranteeing that each object is continuously locked from its first to last access and during the entire processing. An operation $a_i(e)$ of a transaction i is suspended if another transaction j ($i \neq j$) already holds a lock to the object e (and the held lock or the lock to be acquired is an exclusive lock). In this way it is guaranteed that

all conflicting operations of transaction j are executed before those operations of transaction i . When using preclaiming of locks, according to [20] there are no real limits (by determining the required locks before transaction processing in an additional phase), while the occurrence of deadlocks is eliminated, where transactions wait for releasing the locks of each other.

2.3.3 Optimal Transaction Schedules without Blocking. For an example of transactions, see Figure 3, which block each other as in Figure 4. In our example, we want to run these transaction on multi-core CPU with three cores (see Figure 5). In optimal solutions, transactions blocking each other do not run in parallel. We present one optimal solution in Figure 6. We already see that there will be many more optimal solutions. For example, by just exchanging the cores, e.g., core 2 runs the transactions of core 3 in Figure 6 and vice versa, we already get $3! = 6$ equivalent transaction schedules. Hence, we recognize that there will be many optimal solutions, which speeds up Grover’s search algorithm.

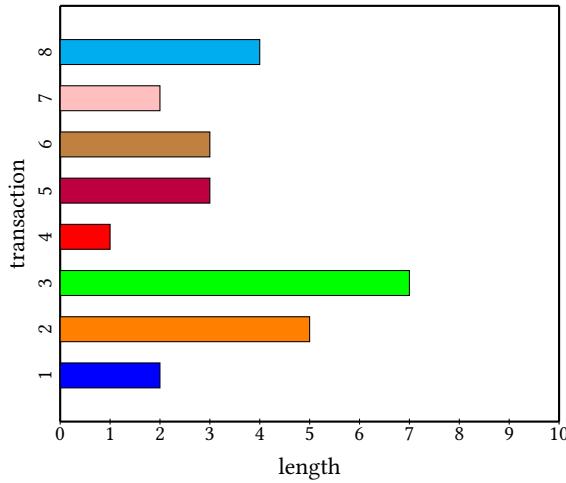


Figure 3: $n = 8$ transactions with their respective lengths

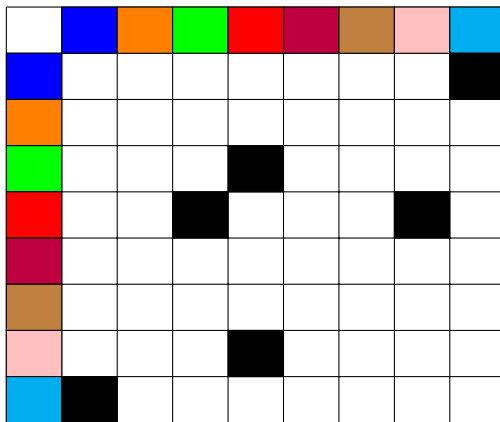


Figure 4: Black fields indicate blocking transactions of the transactions of Figure 3

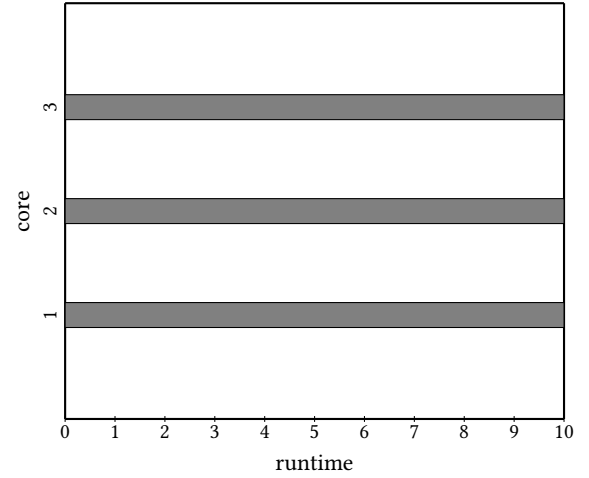


Figure 5: Transactions are scheduled on three cores of a multi-core CPU, the maximum runtime here is 10 time units

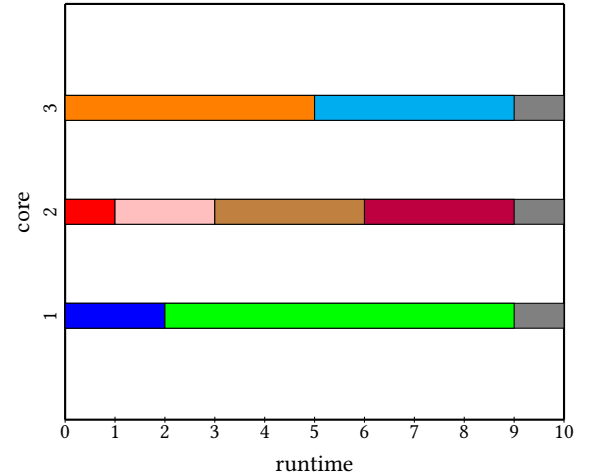


Figure 6: One optimal solution of a transaction schedule for the transactions of Figure 3, which block each other as in Figure 4, on three cores of a multi-core CPU without any blocking with a runtime of 9 time units

3 OPTIMIZING TRANSACTION SCHEDULES BY UNIVERSAL QUANTUM COMPUTERS

We propose the following approach for optimizing transaction schedules on universal quantum computers: The code for the black box function for Grover’s search is generated (see subsection 3.2) based on the transaction lengths and conflicts between the transactions (see Figure 7). Because Grover’s search needs less iterations for those problem instances with more than one solution, we estimate the number of solutions (see subsection 3.3) for our considered transactions in order to speed up our approach. The input of the black box function for Grover’s search is an integer number in superposition of a given number of qubits. In our approach, the input number in superposition represents possible transaction schedules

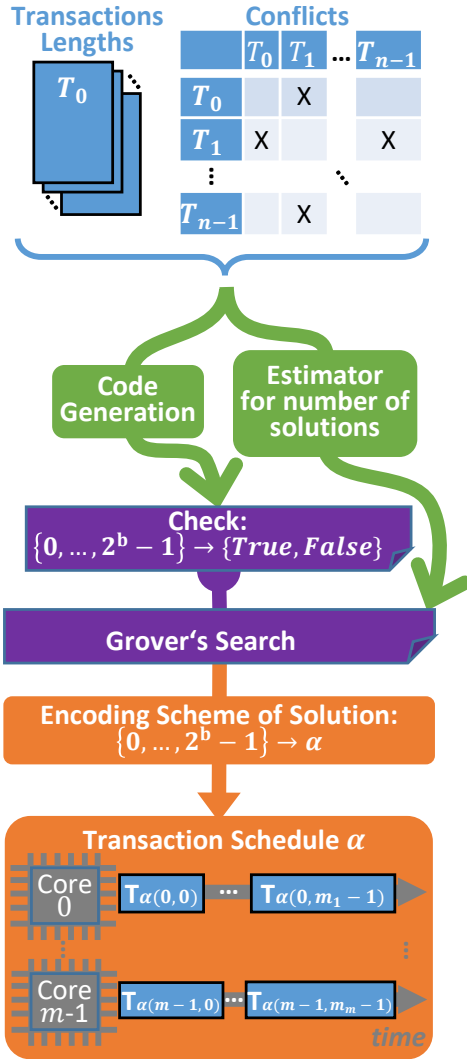


Figure 7: Overview of the code generation approach for optimizing transaction schedules.

and Grover's search returns one integer number of those which successfully passed the black box function. Hence we need a mapping of a given integer number to transaction schedules α , which is also called *encoding scheme* for transaction schedules (see subsection 3.1).

3.1 Encoding Scheme for Transaction Schedules

We describe the representation of a transaction schedule, i.e., the sequence of transactions for each machine (i.e., core of a multi-core CPU), by an integer number and determine the number of used bits.

We propose to split the representation of a transaction schedule into two components: The first component represents a permutation of the given transactions and the second component determines where to split the permutation of transactions into m subsequences to be scheduled to the m machines.

Algo determineSchedule
Input: $p: \{0, \dots, 2^b - 1\}$
Output: $\{0, \dots, n-1\}^{m-1} \times \{0, \dots, n-1\}^{n-1}$

```

for(x in 1..m-1)
     $\mu_x = p \bmod n$ 
     $p = p \div n$ 
a = [0, ..., n-1]
for(i in 0..n-1)
    j = p mod (n-i)
    p = p div (n-i)
     $\pi[i] = a[j]$ 
    a[j] = a[n-i-1]
return ( $\mu_1, \dots, \mu_{m-1}, \pi$ )
    
```

Example (n=4, m=2):
 29

```

x = 1:
 $\mu_1 = 1$ 
p = 7
a = [0, 1, 2, 3]
i = 0: | i = 1: | i = 2: | i = 3:
j = 3 | j = 1 | j = 0 | j = 0
p = 1 | p = 0 | p = 0 | p = 0
 $\pi[0] = 3$  |  $\pi[1] = 1$  |  $\pi[2] = 0$  |  $\pi[3] = 2$ 
a[3]=a[3] | a[1]=a[2] | a[0]=a[1] | a[0]=a[0]
return (1,[3,1,0,2])
    
```

Figure 8: Algorithm for determining a transaction schedule from an integer number with example

Because there are $n!$ possibilities for a permutation π of n items, our domain for representing the permutation of n transactions is $\{0, \dots, n! - 1\}$ for which we need $\lceil \log_2(n!) \rceil$ bits. For decoding $p \in \{0, \dots, n! - 1\}$ to a permutation π of transactions, we use a variant [8] of Lehmer's code [16] with runtime complexity $O(n)$.

Representing splitting a permutation of the n transactions into m subsequences to be scheduled to the m machines, we need $m - 1$ markers μ . We propose to use the domain $\{0, \dots, n-1\}$ for each of the markers and assume that at least one transaction must be processed on the first machine. We require $0 \leq \mu_1 < \mu_2 < \dots < \mu_{m-1} \leq n - 1$, such that otherwise it is not a valid transaction schedule to be discarded by the generated black box function of Grover's search. Defining $\mu_0 := -1$ and $\mu_n := n - 1$, the scheduled transactions on machine m_x are $< \pi(\mu_{m_{x-1}} + 1), \dots, \pi(\mu_{m_x}) >$. The function $\alpha(x, i)$ to determine the i -th transaction on machine x is hence defined as $\alpha(x, i) = \pi(\mu_{m_{x-1}} + 1 + i)$, where $x \in \{0, \dots, m - 1\}$ and $i \in \{0, \dots, \mu_{m_x} - \mu_{m_{x-1}} - 1\}$.

We present the complete algorithm for determining a transaction schedule from an integer number with example in Figure 8.

For each marker we need $\lceil \log_2(n) \rceil$ bits, such that we overall need $\lceil \log_2(n!) \rceil + (m - 1) \cdot \lceil \log_2(n) \rceil$ for the whole representation of the transaction schedule.

3.2 Code Generation Algorithm

The black box function for Grover's search requires qubits as input. Only a limited set of quantum computing operations, i.e., quantum logic gates, can be applied to these qubits. Furthermore, the sequence of quantum computing operations must be fixed in a circuit before the quantum computer starts its processing, i.e., at compile time, such that the quantum computer can be configured accordingly. This is the case for the algorithm for determining a transaction schedule (see Figure 8) whenever the number n of transactions and the number m of machines are fixed at compile time and don't depend on the input, such that e.g. the contained loops can be rolled out already at compile time. Hence the algorithm in Figure 8 can be used within the black box function of Grover's search after changing the input and output types to qubits. Furthermore, due to the limited set of quantum computing operations qubits can e.g.

Algo generateBlackBox

Input: $n: \mathbb{N}$ // number of transactions
 $l: \mathbb{N}^n$ // lengths of transactions
 $m: \mathbb{N}$ // number of machines
 $c: \mathbb{N}$ // number of conflicts
 $o: \{0, \dots, n-1\}^2 \times c$ // conflicts
 $R: \mathbb{N}$ // maximum runtime on each machine

Output: code

“**Algo blackBox**”
Input: $t: \{\lfloor \log_2(n!) \rfloor + (m-1) \times \lfloor \log_2(n) \rfloor\}$ qubits”
Output: {True, False}”
 $(\mu_1, \dots, \mu_{m-1}, \pi) = \text{determineSchedule}(t)$
 $\mu_m = \{n-1\}$
 // Is this schedule valid?
 $\text{result} = \forall i \in \{2, \dots, m-1\}: \mu_{i-1} < \mu_i$
for (i in $0..n-1$)
 $t_{\pi[i]} = \text{switch}(\pi[i])$ // determine length of transaction $\pi[i]$
 for (j in $0..n-1$)
 case $\pi[j]: l[j]$
 determineTimes(-1, 1, $m, \mu_1, \dots, \mu_m, \{s \mid (s, d) \in o \vee (d, s) \in o\}, R)$
for ($(o_1, o_2) \in o$) // Are conflicting transactions overlapping?
 if (**not** ($t_{o_1} \leq t_{o_2}$ or $t_{o_2} \leq t_{o_1}$)) **result** = False”
return result”

Algo determineTimes

Input: $q: \{0, \dots, n-1\}$ // current transaction in π
 $m_c: \{0, \dots, m-1\}$ // currently considered machine
 $m: \mathbb{N}$ // number of machines
 $\mu_1, \dots, \mu_m: \{0, \dots, n-1\}$ // markers for machines
 $os: \text{set}(\{0, \dots, n-1\})$ // set of transactions in conflicts
 $R: \mathbb{N}$ // threshold of max runtime on each machine

Output: code

if ($m_c = m$ or $t_c = n$)
return
for (i in $0..n-1$)
 if ($\mu_{m_c} = i$)”
 $\text{times} = [0, t_{q+1}, t_{q+1} + t_{q+2}, \dots, \sum_{r=q+1}^i t_r]$
 // check runtime of this machine
 if ($\text{times}[\$i+1] > R$) **result** = False”
 // determine start and end time of conflicting transactions
 for (j in $q+1..i$)
 for ($o \in os$)
 if ($\pi[j] == o$)”
 $t_{oa} = \text{times}[\$j-1]$
 $t_{oe} = \text{times}[\$j]$
 + determineTimes($i, m_c+1, m, \mu_1, \dots, \mu_m, os, R$)

Figure 9: Algorithm *generateBlackBox* and its called algorithm *determineTimes* for generating the code of the black box function for Grover’s search. Generated code is represented by string templates, which might contain $\$v$ and $\{\dots\}$ expressions to be replaced with their computed value. For large sets of conflicting transactions, the assignment of start and end times of these transactions can be further improved by using some kind of decision tree (with $O(\log_2(\min(n, c)))$) over the conflicting transactions (for $n \gg c$) or the transaction numbers (for $c \gg n$) for checking if the considered transaction is a conflicting transaction instead of a sequential check.

Algo blackBox

Input: $t: 7$ qubits
Output: {True, False}
 $(\mu_1, \pi) = \text{determineSchedule}(t)$
 $\mu_2 = 2$
 $\text{result} = \mu_1 < \mu_2$
 $t_0 = \text{switch}(\pi[0])$
 case 0: 2
 case 1: 4
 case 2: 1
 $t_1 = \text{switch}(\pi[1])$
 case 0: 2
 case 1: 4
 case 2: 1
 $t_2 = \text{switch}(\pi[2])$
 case 0: 2
 case 1: 4
 case 2: 1
if ($\mu_1 = 0$)
 $\text{times} = [0, t_0]$
 if ($\text{times}[1] > R$)
 result = False
 if ($\pi[0] == 0$)
 $t_{0a} = \text{times}[0]$
 $t_{0e} = \text{times}[1]$
 if ($\pi[0] == 2$)
 $t_{2a} = \text{times}[0]$
 $t_{2e} = \text{times}[1]$
 $\text{times} = [0, t_1, t_2]$
 if ($\text{times}[2] > R$)
 result = False
 if ($\pi[1] == 0$)
 $t_{0a} = \text{times}[0]$
 $t_{0e} = \text{times}[1]$
 if ($\pi[1] == 2$)
 $t_{2a} = \text{times}[0]$
 $t_{2e} = \text{times}[1]$
 if ($\pi[2] == 0$)
 $t_{0a} = \text{times}[1]$
 $t_{0e} = \text{times}[2]$
 if ($\pi[2] == 2$)
 $t_{2a} = \text{times}[1]$
 $t_{2e} = \text{times}[2]$
if ($\mu_1 = 1$)
 $\text{times} = [0, t_0, t_0 + t_1]$
 if ($\text{times}[2] > R$)
 result = False
 if ($\pi[0] == 0$)
 $t_{0a} = \text{times}[0]$
 $t_{0e} = \text{times}[1]$
 if ($\pi[0] == 2$)
 $t_{2a} = \text{times}[0]$
 $t_{2e} = \text{times}[1]$
 if ($\pi[1] == 0$)
 $t_{0a} = \text{times}[1]$
 $t_{0e} = \text{times}[2]$
 if ($\pi[1] == 2$)
 $t_{2a} = \text{times}[1]$
 $t_{2e} = \text{times}[2]$
 $\text{times} = [0, t_2]$
 if ($\text{times}[1] > R$)
 result = False
 if ($\pi[2] == 0$)
 $t_{0a} = \text{times}[0]$
 $t_{0e} = \text{times}[1]$
 if ($\pi[2] == 2$)
 $t_{2a} = \text{times}[0]$
 $t_{2e} = \text{times}[1]$
 if (**not** ($t_{0e} \leq t_{2s}$ or $t_{2e} \leq t_{0s}$))
 result = False
return result

Figure 10: Example of generated black box for $m = 2, n = 3, T = [2, 4, 1]$ and $O = \{(0, 2)\}$ in pseudo code. Some compiler optimizations have already been applied like evaluating constant expressions at compile time and dead code elimination.

neither be used as indices in arrays nor for indirect variable accesses (like in php `$$name`), such that often a huge set of variables holding values in superposition must be hard coded. Hence there is a need for generating code of the black box function for Grover's search for the specific problem instance (here the configuration of transactions including their lengths and conflicts between each other).

We present the overall code generation algorithm in Figure 9 and an example of a generated black box for Grover's search in Figure 10. Please note that the code sometimes is not as elegant as software developers of modern programming languages are used to, but are due to the limited set of quantum computing operations to be arranged in a circuit. The code generation algorithm is open source and publicly available²: We generate code for the Silq [4] quantum computer programming language and simulator, with which we have extensively tested our generated code.

3.3 Estimating the number of solutions k

We already discussed in subsection 2.2 that Grover's search has a speedup with a factor of \sqrt{k} for the number k of solutions. We have run experiments to determine k with 12 transactions and a length of 10 with standard deviation of 3 varying the number of machines and the number of conflicts (see Figure 11) to show the effects in terms of speedups. N is 8,589,934,592 for $m = 2$ and 2,199,023,255,552 for $m = 4$, but already many optimal solutions exist: 48,384,000 (resulting in a speedup of 6,955) for $m = 2$ and 559,872 (resulting in a speedup of 748) for $m = 4$ ($c = 0$). Determining only suboptimal solutions achieves huge speedups: solutions being close to 25% to the optimal solution can be determined with speedups of 38,374 for $m = 2$ and 45,247 for $m = 4$. Hence we highly recommend to run experiments for typical transaction configurations of the used application and based on these experiments estimate the number of solutions for speeding up Grover's search.

3.4 Complexity Analysis

In this section we compare the complexities according to preprocessing, runtime, space and code size of solving the optimizing transaction schedules problem on a traditional computer, universal quantum computer and quantum annealer. Table 1 summarizes the results of the complexities comparison.

3.4.1 Complexity of Algorithm on Traditional Computer. The job shop scheduling problem (JSSP) as simpler variant of the optimizing transaction schedules problem (without considering blocking transactions) is already among the hardest combinatorial optimization problems [10]. Hence the considered algorithm on the traditional computer enumerates all possible transaction schedules: For each transaction schedule it determines the runtimes in $O(n)$ on each machine to determine the minimum execution time and checks that conflicting transactions do not have overlapping execution times in $O(c)$. There are $n!$ possibilities to reorder the transactions (i.e., number of permutations of the n transactions) and the distribution of a permutation of n transactions to m machines is the same as the number of combinations of m objects taken n at a time

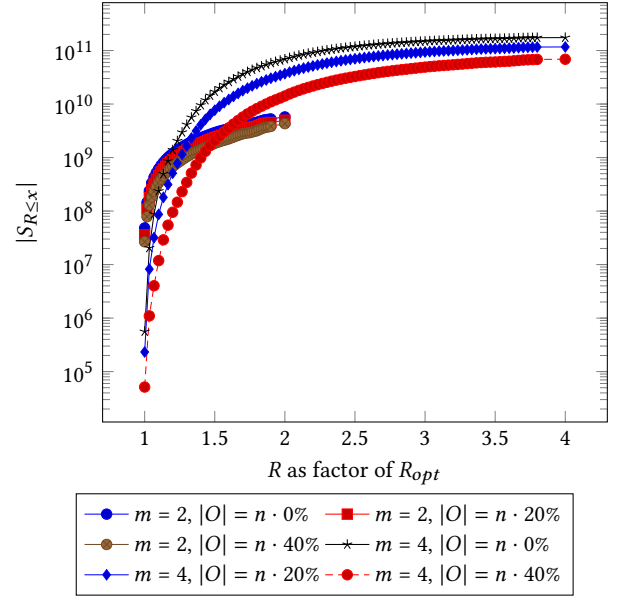


Figure 11: Cumulative number of solutions $|S_{R \le x}|$ in relation to runtime ($n = 12$, $\sigma_{l_i} = 3$).

with repetition, such that $O(n! \cdot \binom{m+n-1}{n})$ possibilities have to be enumerated.

3.4.2 Complexity of Quantum Computing Approach. Grover's search on universal quantum computers takes $O(\sqrt{N})$ iterations, where N is the size of the function's domain and in each iteration the test function needs time $O(n \cdot \log_2(n) + c)$. Please note that determining the length of the i -th transaction in the permutation needs $O(n \cdot \log_2(n))$ time using a decision tree over the transaction numbers. Determining start and end times of conflicting transactions needs $O(n \cdot \log_2(\min(n, c)) + c)$ time by using a decision tree over the conflicting transactions (for $n \gg c$) or the transaction numbers (for $c \gg n$). For our algorithm, $N = O(2^{\log_2(n!) + (m-1) \cdot \log_2(n)}) = O(n! \cdot n^m)$ resulting in an overall runtime complexity of $O(\sqrt{n! \cdot n^m} \cdot (n \cdot \log_2(n) + c))$. The overall runtime complexity can be further improved by estimating the number k of solutions (see subsection 3.3) to $O(\sqrt{\frac{n! \cdot n^m}{k}} \cdot (n \cdot \log_2(n) + c))$, where suboptimal solutions with a higher threshold decreases the runtime further. The space requirement directly depends on the number of used qubits, which is in $O(m \cdot \log_2(n) + \log_2(n!)) = O((n+m) \cdot \log_2(n))$ using Stirling's approximation $n! \sim \sqrt{2 \cdot \pi \cdot n} \cdot (\frac{n}{e})^n$, where \sim means that the two quantities are asymptotic, i.e., their ratio tends to 1 as n tends to infinity.

3.4.3 Complexity of Quantum Annealing Approach. The complexities of the quantum annealing approach have already been determined in [5, 6], but because of simplicity of presentation, the upper bound n^2 for the number c of conflicts have been used in the complexities presented in [5, 6]. Because we directly use the number c of conflicts for Grover's search algorithm in Table 1, we also determine preciser complexities for the quantum annealing approach resulting in $O(m \cdot R^2 \cdot (c \cdot m + n^2))$ for preprocessing time

²<https://github.com/luposdate/OptimizingTransactionSchedulesWithSilq>

	Preprocessing	Execution Time	Space	Code Size
Traditional Computer	$O(1)$	$O(n! \cdot \binom{m+n-1}{n} \cdot (n+c))$ $= O(\frac{(m+n-1)!}{(m-1)!} \cdot (n+c))$	$O(n+c+m)$	$O(1)$
Quantum Computer	$O(n^2 \cdot c)$	$O(\sqrt{\frac{n! \cdot n^m}{k}} \cdot (n \cdot \log_2(n) + c))$	$O((n+m) \cdot \log_2(n))$	$O(n^2 \cdot c)$
Quantum Annealer [5, 6]	$O(m \cdot R^2 \cdot (c \cdot m + n^2))$	$O(1)$	$O(m \cdot R^2 \cdot (c \cdot m + n^2))$	$O(m \cdot R^2 \cdot (c \cdot m + n^2))$

m : number of machines n : number of transactions c : number of conflicts R : maximum execution time on each machine

k : estimated number of solutions

Table 1: Overview preprocessing, runtime, space and code size complexities of the transaction schedule problem for traditional and different types of quantum computers. The space complexity represents the growth of the number of required qubits for quantum computing, number of variables for quantum annealing and number of bytes for traditional computers.

(i.e., time to generate the formula to optimize), space (i.e., number of binary variables) and code size (i.e., size of the formula to optimize). Once initialized, quantum annealers have a constant execution time $O(1)$.

4 SUMMARY AND CONCLUSIONS

In this paper we propose to use Grover’s search for optimizing transaction schedules on universal quantum computers by generating its black box function based on the given problem instance. We describe the encoding scheme for representing a candidate solution and the code generation. We compare the complexities according to preprocessing, runtime, space and code length with approaches for traditional computers and quantum annealers.

In future work, we will investigate improved encoding schemes for optimizing transaction schedules using less bits, other combinatorial optimization problems and also other applications of Grover’s search like quantum cryptography [17].

REFERENCES

- [1] Adriano Barenco, David Deutsch, Artur Ekert, and Richard Jozsa. 1995. Conditional quantum dynamics and logic gates. *Physical Review Letters* 74, 20 (1995).
- [2] William P. Baritompa, David W. Bulger, and Graham R. Wood. 2005. Grover’s quantum algorithm applied to global optimization. *SIAM Journal on Optimization* 15, 4 (2005).
- [3] Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. 1997. Strengths and Weaknesses of Quantum Computing. *SIAM J. Comput.* 26, 5 (1997), 1510–1523. <https://doi.org/10.1137/S0097539796300933>
- [4] Benjamin Bichsel, Maximilian Baader, Timon Gehr, and Martin Vechev. 2020. Silq: A High-Level Quantum Language with Safe Uncomputation and Intuitive Semantics. In *ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)*. 286–300.
- [5] Tim Bittner and Sven Groppe. 2020. Avoiding Blocking by Scheduling Transactions Using Quantum Annealing. In *IDEAS*.
- [6] Tim Bittner and Sven Groppe. 2020. Hardware Accelerating the Optimization of Transaction Schedules via Quantum Annealing by Avoiding Blocking. *Open Journal of Cloud Computing (OJCC)* 7, 1 (2020).
- [7] Michel Boyer, Gilles Brassard, Peter Høyer, and Alain Tapp. 1998. Tight Bounds on Quantum Searching. *Fortschritte der Physik* 46, 4-5 (1998).
- [8] Antoine Comeau. 2014. Mapping between permutations and natural numbers. Blog available at <http://antoinecomeau.blogspot.com/2014/07/mapping-between-permutations-and.html>.
- [9] Thomas M Connolly and Carolyn E Begg. 2005. *Database systems: a practical approach to design, implementation, and management*. Pearson Education.
- [10] M. R. Garey, D. S. Johnson, and Ravi Sethi. 1976. The Complexity of Flowshop and Jobshop Scheduling. *Mathematics of Operations Research* 1, 2 (1976), 117–129.
- [11] Sven Groppe. 2020. Emergent models, frameworks, and hardware technologies for Big data analytics. *The Journal of Supercomputing* 76, 3 (2020), 1800–1827. <https://doi.org/10.1007/s11227-018-2277-x>
- [12] Sven Groppe. 2021. Semantic Hybrid Multi-Model Multi-Platform (SHM3P) Databases. In *International Semantic Intelligence Conference (ISIC 2021), New Delhi (hybrid), India*. <http://ceur-ws.org/Vol-2786/Paper2.pdf>

- [13] Sven Groppe and Jinghua Groppe. 2020. Hybrid Multi-Model Multi-Platform (HM3P) Databases. In *Proceedings of the 9th International Conference on Data Science, Technology and Applications (DATA)*.
- [14] Lov K. Grover. 1996. A Fast Quantum Mechanical Algorithm for Database Search. In *STOC*.
- [15] Lov K. Grover. 1997. Quantum computers can search arbitrarily large databases by a single query. *Physical review letters* 79, 23 (1997).
- [16] Derrick H Lehmer. 1960. Teaching combinatorial tricks to a computer. In *Proc. Sympos. Appl. Math. Combinatorial Analysis*, Vol. 10. 179–193.
- [17] Diana Maimut and Emil Simion. 2018. Post-quantum Cryptography and a (Qu)Bit More. In *SecITC*.
- [18] Kyoichi Okamoto and Osamu Watanabe. 2001. Deterministic Application of Grover’s Quantum Search Algorithm. In *COCOON*.
- [19] Peter W Shor. 1994. Algorithms for quantum computation: Discrete logarithms and factoring. In *Symposium on foundations of computer science*.
- [20] A. Thomson, T. Diamond, S.-C. Weng, K. Ren, P. Shao, and D. J. Abadi. 2012. Calvin: Fast Distributed Transactions for Partitioned Database Systems. In *SIGMOD*.

Customized Eager-Lazy Data Cleansing for Satisfactory Big Data Veracity

Rim Moussa

rim.moussa@enicarthage.rnu.tn
University of Carthage
Tunisia

Soror Sahri

soror.sahri@parisdescartes.fr
University of Paris
France

ABSTRACT

Big data systems are becoming mainstream for big data management either for batch processing or real-time processing. In order to extract insights from data, quality issues are very important to address, particularly. A veracity assessment model is consequently needed. In this paper, we propose a model which ties quality of datasets and quality of query resultsets. We particularly examine quality issues raised by a given dataset, order attributes along their fitness for use and correlate veracity metrics to business queries. We validate our work using the open dataset *NYC taxi' trips*.

CCS CONCEPTS

• **Information systems** → *Query reformulation; Integrity checking; Data cleaning.*

KEYWORDS

Veracity, Big data,

ACM Reference Format:

Rim Moussa and Soror Sahri. 2021. Customized Eager-Lazy Data Cleansing for Satisfactory Big Data Veracity. In *25th International Database Engineering Applications Symposium (IDEAS 2021)*, July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 9 pages. <https://doi.org/10.1145/3472163.3472195>

1 INTRODUCTION

CrowdFlower surveyed over hundred fifty (153) data scientists nationwide between November 2014 and December 2014 [4]. 52.3% of data scientists cited poor quality data as their biggest daily obstacle, and 66.7% of data scientists said cleaning and organizing data is their most time-consuming task. The unprecedented scale at which data is generated today has shown a large demand for scalable data collection, data cleansing, and data management. *Big data* features characteristics known as 5 V's: *volume*, *velocity*, *variety*, *veracity* and *value*. *Volume* refers to the amount of data which henceforth increased to the range of Tera and Peta Bytes scale; *velocity* refers to the speed at which new data is generated and processed, henceforth the challenge is to analyze data while it is being generated; and *variety* refers to different types of data; e.g. structured (relational data),

semi-structured (XML, JSON), unstructured (text, image, video). While the three first V's are well-defined, *Veracity* and *Value* are not well-defined, and consequently hard to measure. Moreover, if the data is inaccurate, imprecise, incomplete, inconsistent or has uncertain provenance, then any insights obtained from data analytic are meaningless and unreliable to some extent. Data scientists conduct data analysis with, e.g., incomplete data, and provide error bars to reflect the uncertainty in the results of the analysis. They obtain errors' snowball in case of big impacts of errors and butterfly effect in case of small errors impacts [15]. Hereafter, we present the motivations of our research,

- *Absence of Standards and Norms*: there is still no consensus on what the notion of *Veracity* is, and consequently no standard approach for measurement of *veracity*.
- *Disconnect between data source and data use*: Data users and data providers are often different organizations with very different goals and operational procedures. In many cases, the data providers have no idea about the business use cases of data users. This disconnect between data source and data use is one of the prime reasons behind the data quality issues.
- *Big data and Distributed Architectures*: Processing Big Data requires highly scalable data management systems. Performance aspects should not be overlooked. In distributed data management systems, the quality issues are particularly difficult due to (i) large datasets distributed across multiple nodes, (ii) query processing involves multiple nodes, and (iii) new data batches and real-time data alter the database and consequently quality and veracity metrics become stale.
- *Schema-on-read and the move from traditional Data Warehouse Systems To Data Lakes and Lakehouses' Architectures* [17]: data lakes and lakehouses are a "catch all" repositories, where *schema-on-read* doesn't impose a specific model for all datasets. Consequently, problems such as data quality, provenance, and governance, which have been historically associated with traditional data warehouses are more complex in a data lake or lakehouse system architecture [14].

In this paper, we propose a suitable model dealing with Veracity characteristic assessment. We come up with a model which ties quality of datasets and veracity of query resultsets in a distributed data store. In our framework *data quality* is defined as being the fitness for use of data, our model details how the veracity metrics on a given distributed dataset are calculated, and fits the needs and requirements of users' queries. It particularly orders attributes along their fitness for use and correlate veracity metrics to business queries. We validate our work using an open dataset *NYC cabs' trips*. The latter contains over a billion of individual taxi trips in the city of New York from January 2009 to present.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](https://permissions.acm.org).

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Association for Computing Machinery.

ACM ISBN 978-1-4503-8991-4/21/07...\$15.00

<https://doi.org/10.1145/3472163.3472195>

In the rest of the paper, we investigate veracity metrics calculus in Section 2. Then, we validate our proposed veracity framework using the NYC cabs dataset in Section 3. Section 4 overviews related work. Finally, Section 5 recalls the contributions and outlines our future work.

2 VERACITY FRAMEWORK

In this section, we investigate quality and veracity metrics calculus for a big dataset and a given workload. In our model, we propose taking into account all of data inconsistency, data inaccuracy, and data incompleteness without data repairing them and without dropping rows from the original dataset. Indeed, *the same data can fit for one business query, but not for another*. We consider different cleansing strategies for different queries. We recall that data cleansing process is complex and consists of several stages which include specifying the quality rules, detecting data error and repairing the errors. Most of data cleansing approaches undertake a data repair action to remove errors from data sources. However, in real-world applications, cleansing the data is costly, and may lead to a loss of potentially useful data. Moreover, the existing cleansing approaches cannot ensure the accuracy of the repaired data and require domain expert to validate repair rules. In close related work the consistent query answering is ensured without modifying sources [3]. These works focus on integrity constraints to express data quality rules, as functional dependencies or patterns (ex. Conditional functional dependencies). Such rules can capture a wide variety of errors including duplication, inconsistency, and missing values. However, consistent query answering was only investigated for relational data and not for *Schema-on-read* and big data datasets.

2.1 Data Metrics Model

We assume that data is in row format. Each row describes a record with attributes. We overview how some quality metrics are computed by considering the characteristics of distributed data, in particular their fragmentation and replication characteristics. For this, let \mathcal{R} be a set of datasets $\{R_i\}$. Attributes $\{A_{ij}\}$ describe each dataset R_i .

2.1.1 Data Consistency. It expresses the degree to which a set of data satisfies a set of integrity constraints (e.g. check constraints, reference constraints, entity constraints and functional dependencies). Let,

- C be the set of constraints $\{C_{ij}\}$, such that $\{C_{ij}\}$ relate to dataset R_i . Constraints' types are functional dependencies, check constraint, reference constraint, entity constraint. A constraint is defined over a single attribute (e.g. domain definition, unique, and so on) or over multiple attributes (e.g. functional dependency, measure expression).

Data consistency metric counts, for each dataset $\{R_i\}$ and each attribute $\{A_{ij}\}$, the number of rows which violate any of the defined constraints $\{C_{ij}\}$.

- **Data Fragmentation:** If the dataset is fragmented horizontally the calculus of the data consistency of dataset $\{R_i\}$ is first performed for each horizontal partition, then all counts are summed up.
If the dataset is fragmented vertically the calculus of the

data consistency of dataset $\{R_i\}$ is first performed for each vertical partition for each attribute, the overall counts results are obtained with union all.

If the dataset is fragmented horizontally and vertically (i.e. hybrid fragmentation), the calculus of the data consistency of dataset $\{R_i\}$ is first performed through vertical partitions, then horizontal partitions.

- **Data Replication:** The refresh protocol is very important. Indeed, some protocols are strict and others are not. Most of analytical systems fall into BASE systems category (i.e. Basically Available, Soft state, Eventual consistency). Hence, in the case of master-slave replication, for each replica we propose the calculation of a consistency score. In the case of peer-to-peer replication, following each refresh operation (update, insert or delete propagation), we propose to calculate a *refresh propagation ratio* at each node, stated as *number of not refreshed replicas / number of replicas*.

2.1.2 Data Completeness. It relates to the presence and absence of features, their attributes and relationships. Data completeness metric counts, for each dataset $\{R_i\}$ and each attribute $\{A_{ij}\}$, the number of null values (i.e. missing values).

- **Data Fragmentation:** If the dataset is fragmented horizontally the calculus of the data completeness of dataset $\{R_i\}$ is first performed for each horizontal partition and for each attribute, then all counts are summed up.
If the dataset is fragmented vertically the calculus of the data completeness of dataset $\{R_i\}$ is first performed for each vertical partition, the overall counts are obtained with union all.
If the dataset is fragmented horizontally and vertically (i.e. hybrid fragmentation), the calculus of the data completeness of dataset $\{R_i\}$ is first performed through vertical partitions, then horizontal partitions.
- **Data Replication:** The data completeness metric is to be calculated for each replica.

2.1.3 Data Accuracy. In [16], accuracy is defined as *the extent to which data are correct, reliable and certified*. It introduces the idea of how precise, valid and error-free is data. There are three main accuracy definitions in the literature: (i) *semantic correctness* that describes how well data represent states of the real-world [16]; (ii) *syntactic correctness* that expresses the degree to which data is free of syntactic errors such as misspellings and format discordances [11]; and (iii) *precision* that concerns the level of detail of data representation [13][12]. The ISO 19113 standard on geospatial data recommends the use of (1) *positional accuracy* -accuracy of the position features, (2) *temporal accuracy* -accuracy of the temporal attributes and temporal relationships of features and (3) *thematic accuracy* -accuracy of quantitative attributes and the correctness of non-quantitative attributes and of the classifications of features and their relationships. Data accuracy metric counts, for each dataset $\{R_i\}$ and each attribute $\{A_{ij}\}$, the number of inaccurate values.

- **Data Fragmentation:** If the dataset is fragmented horizontally the calculus of the data accuracy of dataset $\{R_i\}$ is first performed for each horizontal partition and for each attribute,

then all counts are summed up.

If the dataset is fragmented vertically the calculus of the data accuracy of dataset $\{R_i\}$ is first performed for each vertical partition, the overall counts are obtained with union all. If the dataset is fragmented horizontally and vertically (i.e. hybrid fragmentation), the calculus of the data accuracy of dataset $\{R_i\}$ is first performed through vertical partitions, then horizontal partitions.

- **Data Replication:** The data accuracy metric is to be calculated for each replica.

2.1.4 Data Timeliness/Freshness. *Data freshness* introduces the idea of how old is the data: Is it fresh enough with respect to the user expectations? Has a given data source the more recent data? *Timeliness* [16] describes how old is data (since its creation/update at the sources). It captures the gap between data creation/update and data delivery.

2.1.5 Data Accessibility. *Accessibility* denotes the ease with which the user can obtain the data analyzed (cost, timeframe, format, confidentiality, respect of recognized standards, ...).

2.2 Query Metrics Model

Our query model computes veracity metrics of the query resultsets on a given distributed dataset, based on the quality metrics model detailed in §2.1. For simplicity and without loss of generality, we assume that the quality metrics of the dataset are already defined by considering the distributed data characteristics (fragmentation and replication). We consequently consider for the rest of the paper, the distributed scenario where data is collected by a single entity, but stored and processed in a distributed manner for scalable query processing.

Let's consider the following notations: Given a big dataset D , let $\{Q_i\}$ for $i \in 1, \dots, n$, be the set of queries on D , such that,

- Each query Q_i relates to a user profile, and each profile is described by multiple queries,
- Each query Q_i involves a set of attributes $\{A_j\}$, with $j \in 1, \dots, m$. For example Q_1 involves attribute set $\{A_1, A_2\}$. Notice that a set of attributes can appear or not for different queries of different user profiles. That is each attribute can be relevant (or not) to each query and then to each user profile.
- For each attribute set involved in Q_i , there is a quality constraint C_j . For example, The constraint C_1 corresponds to a completeness constraint on attribute A_1 ; and the constraint C_2 corresponds to a consistency constraint on $\{A_1, A_2\}$.
- Each quality constraint C_j has a noise N_j that corresponds to the percent of rows in the dataset D which violate C_j .

We propose a measure named *veracity score* (denoted *V-score*) which correlates the data quality metrics to the quality of queries' resultsets. The veracity score is intended to capture for each query Q_i , how much the noise through a quality constraint C_j affects the veracity of its resultset and then the fitness for use of the attributes. The noise implied by each quality constraint can increase or decrease the *V-score* according to a defined *cleansing approach*. Our query model implements three cleansing approaches: *eager*, *custom* and *lazy* approaches. We recall that the cleansing in our work is

performed at the query answering and not at the data sources. In the sequel, we describe the three cleansing approaches,

- (1) **Eager virtual data cleansing:** All rows in the dataset must satisfy all defined quality constraints. Hence, none of the quality constraints is violated in the cleansed dataset, and all the data metrics are satisfactory. Indeed, if a consistency constraint states that $A_i \geq 0$, is violated, then all rows where $A_i < 0$ are not considered in the query result whether the query uses the attribute A_i , or not. This approach may discard rows of interest for queries, leading to a false query answer set. For that reason, we assume that every row deleted will affect the quality of the answer.
The noise is considered with positive impact for all quality constraints satisfied while the query is not using their involved attributes; and vice versa.
- (2) **Custom virtual data cleansing:** Quality constraints are checked if and only if the attributes within the constraints are used by the query (i.e. $U(Q_i, A_i) = 1$). Hence, we discard all rows which violate any constraint involving A_i attribute.
- (3) **Lazy virtual data cleansing:** none of the constraints is checked. Each query executes on the dataset as it is. This approach aims at saving filters' execution. Notice that, the custom-cleansing approach might outperform this approach, since it runs over a filtered dataset.

Summarizing, for a given query, custom-cleansing corresponds to the highest veracity score, while eager and lazy cleansing correspond to the lowest veracity score. The I/O cost affects query performance. Indeed, vertical fragmentation implemented by wide-column data stores reduces the number of read operations and transfers only required data to main memory for both custom and lazy data cleansing. Likewise data filtering -enabled by constraints expressions for eager and customer cleansing outputs a small dataset performed prior to complex operations such as joins and grouping will also improve performance. We assume that attributes and constraints are equal in veracity importance assessment. The model can be extended with custom weights assigned to constraints. We propose the formulas shown in Equation 1 and Equation 2 to calculate the veracity of a query with respect to a quality constraint, and for a selected *approach*.

$$V - \text{Score}_{ij}(Q_i, C_j) = W_{\text{approach}} \times N_{C_j} \quad (1)$$

$$V - \text{Score}(Q_i) = \sum_{j=1}^m (V - \text{Score}_{ij}(Q_i, C_j)) \quad (2)$$

As explained earlier, we consider three approaches, consequently *approach* is either *eager*, *custom*, or *lazy*. N_{C_j} denotes the noise that corresponds to quality constraint C_j . We propose the following rule for assessing the veracity of each query with respect to each cleansing approach: If the query Q_i doesn't use attributes of the constraint C_j or uses them with no impact on Q_i 's answer set correctness; Then $W_{\text{eager}} = -1$ (Eager approach applies constraints on data not retrieved for analysis, and this may alter the veracity of the query answer set), $W_{\text{custom}} = 0$ (Custom approach does not apply constraints on data not retrieved by the query), and $W_{\text{lazy}} = 0$ (Lazy approach does not apply constraints on data not retrieved by the query); Else $W_{\text{eager}} = +1$ (Eager approach applies constraints on data retrieved by the query), $W_{\text{custom}} = +1$ (Customer approach

applies constraints on data retrieved by the query), and $W_{lazy} = -1$ (Lazy approach does not apply constraints on data retrieved for analysis, and this may alter the veracity of the query answer set).

3 USE CASE: NYC TAXI DATASET

In this Section, we consider the NYC taxi dataset to validate our proposed model. We first present this dataset and related data quality metrics. Then, we accordingly evaluate the quality of taxi trips query answer set using motivating examples.

Information associated with cabs' trips can provide unprecedented insight into many different aspects of city life, from economic activity and human behavior to mobility patterns. But analyzing these data presents many challenges. The data are complex, containing geographical and temporal data in addition to multiple variables associated with each trip [5] [10]. The *New York City Taxi and Limousine Commission* has released a detailed historical dataset covering over a billion individual taxi trips in the city from January 2009. Each individual trip record contains precise location coordinates for where the trip started and ended (spatial data), timestamps for when the trip started and ended (temporal data), plus a few other attributes including fare amount, payment method, and distance traveled. We calculate other attributes such as *geohash-pickup*, *geohash-dropoff*, *trip-duration (min)*, *average-speed (mph)* as well as the day-time hierarchy: *year* → *month* → *day of week* → *hour*.

3.1 Quality Metrics of The NYC Taxi Data

3.1.1 Data consistency. We propose the following constraints for cabs' mobility data of New York City dataset,

- **Spatial constraints:** The spatial data in the dataset relate to pick-up and drop-off longitude and latitude. Several records feature invalid longitude and latitude values -longitude range is $-180^\circ..180^\circ$, and latitude range is $-90^\circ..90^\circ$, are valid but not in the envelope of New York City. Moreover, maps' plots show also cabs on the sea [10]. For 2015 yellow cabs data collection, over two millions and half of trips (1.75%) relate to not valid GPS coordinates (2,555,473 out of 146,112,989). GPS coordinates are checked whether they fall in New York City Minimum Bounding Box (MBB) or not (see first constraint in Table 1).
- **Temporal constraints:** Each file groups cabs' trips of a given month. Consequently date-time columns in the dataset should conform to the file information. Trips' durations must be positive. For yellow cabs trips in 2015, 1,215,064 trips out of 146,112,989 trips feature a duration less or equal to 0 minutes, i.e. 0.83% (i.e. $\text{pickup-timestamp} > \text{dropoff-timestamp}$).
- **Domain constraints:** The maximum number of passengers is 5. For yellow cabs trips in 2015, 5,124,540 trips out of 146,112,989 trips feature a number of passengers greater than 5, i.e. 3.5%.

The records with trip distance less than half a mile or greater than 50 miles are not valid. For trips in 2015, 6,479,521 trips out of 146,112,989 trips feature an erroneous trip distance, i.e., 4.43%.

Fare Amount: Some records have either a very high value of the fare amount or negative values. These records give

erroneous min, max and average values for estimating the cost of a trip. The percentage of negative or zero amounts is equal to 0.004% (65,301 trips)

3.1.2 Data completeness. Yellow cabs trips reported in 2015 feature all data. In our framework, we propose to calculate for each attribute a noise value, as illustrated in Table 1. Then, for each constraint, we propose to compute an incidence matrice, to show whether the query uses or not the attribute set involved in the constraint definition (see Table 2). In Table 3, we detail how to measure the fitness for use of each approach, as well as approaches' ranking.

3.2 User profiles and business queries

User profiles for cabs' mobility data analysis fall into the following groups,

- Social scientists who study cabs' passengers' preferences. Examples of queries include where cabs go on Friday and Saturday evening?
- City planners who focus on roads' capacities, tolls' fees. Examples of queries include traffic jam detection.
- Cabs' dispatchers who optimize cabs dispatching for maximizing profit. Examples of queries include: identify where cabs should be at a given time (day, hour, pickup location); or Compare trips with tolls fees versus trips without tolls fees with same pattern from-to and same time-interval
- Customers who want to know min-max duration and cost of trips from one location to another. Examples of queries include average, minimum, maximum cost as well as average, minimum, maximum duration of a trip from a location to another on a given day and time.

For each query, the focus will be put on whether quality metrics (the accuracy of spatial data, the completeness of data and the consistency) impact the veracity of query answer set or not.

3.3 Workload Examples

Next, we detail three queries, namely Q1, Q2 and Q3. For each query, we give its SQL statement and its results for different cleansing approaches respectively *lazy cleansing*, *custom cleansing* and *eager cleansing*. *Google Big query* shows the query processing time as well as the volume of data used for data processing. Notice that the *Google Big query* is an immutable, column-oriented store, and the more check constraints are enabled the more data is retrieved for calculating the query answer. Indeed, check constraints apply to other attributes.

Query 1 calculates Top K frequent trip patterns (*geohash-pickup* - *geohash-dropoff*) (see Figures 1, 2). After discarding trips with *geohash-pickup* and *geohash-dropoff* that are not in the valid longitude and latitude range, the resultset appears without 214,331 misleading trips (when enabling WHERE Clause in Figure 1). Hence, *eager cleansing* alters query resultset, degrades query performance, and presents a different result than a *lazy cleansing*. Notice that the elapsed time of the query with *lazy cleansing* is 5sec while the query with the WHERE clause is 6.7sec (*eager cleansing*). Also, the volume of retrieved data is 4.4GB for *lazy* and *custom cleansing* versus 10.9GB for *eager cleansing*. This confirms that wide-column data stores (such as BigQuery) is optimal for *lazy* and customer

Attribute set	Constraints	Noise (%)
{latitude, longitude} for pickup and dropoff	$C_1: 40.491603 \leq \text{latitude} \leq 45.01785$ and $-79.762152 \leq \text{longitude} \leq -71.856214$	1.75
{pickup-timestamp, dropoff-timestamp}	$C_2: \text{dropoff-timestamp} > \text{pickup-timestamp}$	0.83
{trip-distance}	$C_3: 0.5 \text{ mi} < \text{trip-distance} < 40 \text{ mi}$	4.43
{nbr-of-passengers}	$C_4: 1 \leq \text{nbr-of-passengers} \leq 5$	3.5
{fare-amount}	$C_5: \text{fare-amount} > 0$	0.004
{tolls-amount}	$C_6: \text{tolls-amount} \geq 0$	0.0002

Table 1: Noise degree of each attribute.

Constraint	Q1	Q2	Q3
C_1	1	1	1
C_2	0	1	1
C_3	0	0	0
C_4	0	0	0
C_5	0	1	1
C_6	0	1	0

Table 2: Incidence Matrice (constraint-Query).

cleansing, while it isn't for eager cleansing. The latter performs all constraints check and retrieves consequently more attributes' data. We notice that the answer set on the raw data (lazy cleansing), and the answers set is different for the three approaches. Indeed, enabling all constraints on geography, cost, time, trip attributes with incorrect values can affect the consistency of the answer set. The custom cleansing approach is preferable for this query.

Query 2 Compares the cost and duration of trips for a given hour and day from a pick-up geographic zone to a drop-off geographic zone with and without tolls (see Figure 3). The answer set of this query is illustrated in Figure 4, and is the same for the three approaches. But, the volume of retrieved data is different, we obtain 8.7GB for lazy approach, 9.8GB, and 12GB for eager approach with all constraints enabled. The lazy cleansing approach is then preferable for this query.

Query 3 calculates the following measures AVG, MIN, MAX over cost and duration of trips for given hour and day from a pickup geographic zone to a drop-off geographic zone (see Figure 5). We notice that the resultset on the raw data can have many errors in terms of accuracy and consistency (see Figure 6). Indeed, entries (e.g. time and cost attributes) with incorrect values (negative or zero values) can affect the computation of trip duration and trip cost, and then the consistency of the answer set. The data volume retrieved for lazy and custom approaches is 7.6GB, and is 10.9GB for the eager approach enabling all constraints verification. The customer cleansing approach is then preferable for this query.

4 RELATED WORK

In this Section, we overview related work which tackle data quality issues in general, as well as research papers which investigate data quality issues in particular domains. In [7], Fletcher proposes a framework that identifies data quality (DQ) problems in distributed computing systems (DCS), and how the relationship between data quality and DCS can impact decision making and organizational

performance. To present this relationship, Fletcher proposes crossing in a matrix DCS attributes (e.g. data replication, partitioning, etc) to DQ dimensions (e.g. accuracy, timeliness, etc). A cell in the matrix identifies a possible relationship between the DQ dimension and the DCS attribute. For example, the *Accuracy-Data Replication* cell indicates that accuracy would be impacted by data replication. The objective of [2] is to find dynamically the best trade-off between the cost of the query and the quality of the result retrieved from several distributed sources. For this purpose, Berti-Equille proposes a query processing framework for selecting dynamically sources with quality-extended queries with a negotiation strategy. The idea behind is merging the quality of service approach together with the quality of data techniques. From the quality of service domain, the author extends QML (quality of service modeling language) a query language for describing and manipulating data and source quality contracts (ex. accuracy, completeness, freshness and consistency). The result is a quality-extended query language (XQual) that presents in a flexible way, the specification of quality requirements. Saha and Srivastava [15] present challenges of data quality management that arise due to volume, velocity and variety of data in the Big Data era. They present two major dimensions of big data quality management, (i) discovering/learning based on data and (ii) accuracy vs efficiency trade-off under various computing models (centralized and distributed computing models).

Batini et al. [1] compare thirteen methodologies for data quality assessment and improvement along several dimensions. The comparison dimensions includes (i) the methodological phases and steps that compose the methodology, (ii) the strategies and techniques that are adopted in the methodology for assessing and improving data quality levels, (iii) the data quality dimensions and metrics that are chosen in the methodology to assess data quality levels, (iv) types of costs that are associated with data quality issues (indirect costs associated with poor data quality and direct costs of assessment and improvement activities), (v) types of data, (vi) types of information systems, (vii) organizations involved in data management, (viii) processes that create or update data with the goal of producing services required by users that are considered in the methodology and (ix) the services that are produced by the processes that are considered in the methodology.

In the literature, the following projects conducted on real datasets tackle data quality issues,

- *PEDSnet*: a clinical data research network aggregates electronic health record data from multiple children hospitals

Table 3: Running Example for calculating the fitness for use of each approach. Each triplet(x,y,z) corresponds respectively to the v -score of each approach eager, custom, and lazy.

	C_1	C_2	C_3	C_4	C_5	C_6	$FU_{Q_i, approach}$	Approaches' ranking
Q_1	1.75	-0.83	-4.43	-3.5	-0.004	-0.0002	-7.0142	custom, lazy, eager
	1.75	0	0	0	0	0	1.75	
	-1.75	0	0	0	0	0	-1.75	
Q_2	1.75	0.83	-4.43	-3.5	0.004	0.0002	-5.3458	custom, lazy, eager
	1.75	0.83	0	0	0.004	0.0002	2.5842	
	-1.75	-0.83	0	0	-0.004	-0.0002	-2.5842	
Q_3	1.75	0.83	-4.43	-3.5	0.004	-0.0002	-6.3462	custom, lazy, eager
	1.75	0.83	0	0	0.004	0	1.584	
	-1.75	-0.83	0	0	-0.004	0	-1.584	

```

SELECT ST_GEOHASH(ST_GEOPOINT(pickup_latitude , pickup_longitude ),6) AS pickup_geohash ,
ST_GEOHASH(ST_GEOPOINT(dropoff_latitude , dropoff_longitude ),6) AS dropoff_geohash ,
COUNT(*) count_trips
FROM bigquery-public-data.new_york.tlc_yellow_trips_2015
WHERE (pickup_latitude between -90 and 90)
AND (dropoff_latitude between -90 and 90)
AND (pickup_longitude between -180 and 180)
AND (dropoff_longitude between -180 and 180)
GROUP BY pickup_geohash , dropoff_geohash
ORDER BY count_trips DESC
LIMIT 10

```

Figure 1: Q1: Top 10 frequent pickup-dropoff locations, with none of the constraints in Table 1 enabled.

to enable large-scale research, and is presented in [8, 9]. Authors propose a set of systematic data quality checks to be implemented and executed on the data.

- *ISMIR*: Besson et al. examine quality issues raised by the development of XML-based Digital Score Libraries and propose a quality management model.
- In [6] Firmani et al. choose a specific instance of such a type (notably deep Web data, sensor-generated data, and Twitters/short texts) and discuss how quality dimensions can be defined in these cases.

5 CONCLUSION

Summarizing, in this paper, we overview related work on data quality assessment and motivate the consideration of different cleansing strategies appropriate for each business query on immutable and wide-column data store (Google BigQuery) where data is queried by different profiles and conflicting queries. We show that it is not appropriate to delete rows which violate quality constraints. For each query, one should compare the answer set for the three cleansing approaches. Future work is devoted to data profiling and preparation for adequate querying by the query engine in big data lakes and lakehouses' architectures.

REFERENCES

- [1] Carlo Batini, Anisa Rula, Monica Scannapieco, and Gianluigi Viscusi. 2015. From Data Quality to Big Data Quality. *J. Database Manag.* 26, 1 (2015), 60–82.

- [2] Laure Berti-Equille. [n.d.]. Quality-Adaptive Query Processing Over Distributed Sources. In *9th International Conference on Information Quality (IQ)*.
- [3] Leopoldo Bertossi. 2006. Consistent Query Answering in Databases. *SIGMOD Rec.* 35, 2 (2006), 68–76.
- [4] CrowdFlower. 2015. Data Scientist Report. https://visit.figure-eight.com/rs/416-ZBE-142/images/Crowdflower_Data_Scientist_Survey2015.pdf. Accessed = 2021-06-01.
- [5] Nivan Ferreira, Jorge Poco, Huy T. Vo, Juliana Freire, and Claudio T. Silva. 2013. Visual exploration of big spatio-temporal urban data: A study of New York city taxi trips. *IEEE Transactions on Visualization and Computer Graphics* 19, 12 (2013), 2149–2158.
- [6] Donatella Firmani, Massimo Mecella, Monica Scannapieco, and Carlo Batini. 2016. On the Meaningfulness of "Big Data Quality" (Invited Paper). *Data Science and Engineering* 1, 1 (2016), 6–20.
- [7] Finbar Fletcher. 1998. *A Framework For Addressing Data Quality in Distributed Systems*. Technical Report.
- [8] Ritu Khare, Levon Utidjian, Byron J. Ruth, Michael G. Kahn, Evanette Burrows, Keith Marsolo, Nandan Patibandla, Hanieh Razzaghi, Ryan Colvin, Daksha Ranade, Melody Kitzmiller, Daniel Eckrich, and L. Charles Bailey. 2017. A longitudinal analysis of data quality in a large pediatric data research network. *JAMIA* 24, 6 (2017), 1072–1079.
- [9] Ritu Khare, Levon Utidjian, Gregory Schulte, Keith Marsolo, and Charles Bailey. 2015. Identifying and Understanding Data Quality Issues in a Pediatric Distributed Research Network. In *American Medical Informatics Association Annual Symposium (AMIA)*.
- [10] Rim Moussa, Ahmed Haddad, and Tarek Bejaoui. 2018. The Quest for Scalable and Intelligent Trajectory Data Analytics Systems: Status Report and Future Directions. In *International Conference on Smart Communications and Networking SmartNets*. 1–6.
- [11] Felix Naumann, Ulf Leser, and Johann Christoph Freytag. 1999. Quality-driven Integration of Heterogenous Information Systems. In *Proceedings of the 25th International Conference on Very Large Data Bases*.
- [12] Jack E. Olson. 2003. *Data Quality: The Accuracy Dimension* (1st edition ed.). Morgan Kaufmann Publishers Inc.
- [13] Thomas C. Redman. 1996. *Data quality for the information age*. Artech House.

Row	pickup_geohash	dropoff_geohash	count_trips
1	7zzzzz	7zzzzz	1970930
2	dr5rud	dr5ru6	331095
3	dr5ru7	dr5rus	320923
4	dr5rue	dr5ru6	320594
5	dr5rus	dr5ru6	316013
6	dr5ru6	dr5rus	292426
7	dr5ru6	dr5rud	287102
8	dr5ru7	dr5ru6	284685
9	dr5ru6	dr5rue	282201
10	dr5ru7	dr5rue	272592

Row	pickup_geohash	dropoff_geohash	count_trips
1	dr5rud	dr5ru6	331095
2	dr5ru7	dr5rus	320923
3	dr5rue	dr5ru6	320594
4	dr5rus	dr5ru6	316013
5	dr5ru6	dr5rus	292426
6	dr5ru6	dr5rud	287102
7	dr5ru7	dr5ru6	284685
8	dr5ru6	dr5rue	282201
9	dr5ru7	dr5rue	272592
10	dr5rue	dr5rus	272588

Row	pickup_geohash	dropoff_geohash	count_trips
1	dr5rud	dr5ru6	307095
2	dr5rus	dr5ru6	304586
3	dr5rue	dr5ru6	301115
4	dr5ru7	dr5rus	296468
5	dr5ru6	dr5rus	281084
6	dr5ru6	dr5rue	268055
7	dr5ru6	dr5rud	262563
8	dr5ruk	dr5ru6	253985
9	dr5ru7	dr5rue	249938
10	dr5rud	dr5rus	241256

Figure 2: Q1: Top 10 trips per pick-up and drop-off geohashes answers set for respectively lazy (upper-left), custom (upper-right) and eager (lower-middle) virtual cleansing approaches.

- [14] Janessa Rivera and Rob van der Meulen. 2014. Gartner Says Beware of the Data Lake Fallacy. <https://www.gartner.com/en/newsroom/press-releases/2014-07-28-gartner-says-beware-of-the-data-lake-fallacy>. Accessed = 2021-06-01.
- [15] Barna Saha and Divesh Srivastava. 2014. Data quality: The other face of Big Data. In *IEEE 30th International Conference on Data Engineering*. 1294–1297.
- [16] Richard Y. Wang and Diane M. Strong. 1996. Beyond Accuracy: What Data Quality Means to Data Consumers. <http://www.jmis-web.org/articles/1002>. *J. of Management Information Systems* 12, 4 (1996), 5–33.
- [17] Matei Zaharia, Ali Ghodsi, Reynold Xin, and Michael Armbrust. 2021. Lakehouse: A New Generation of Open Platforms that Unify Data Warehousing and Advanced Analytics. In *11th Conference on Innovative Data Systems Research, CIDR*. http://cidrdb.org/cidr2021/papers/cidr2021_paper17.pdf


```

WITH trips AS (SELECT *,
EXTRACT(HOUR FROM pickup_datetime AT TIME ZONE "America/New_York") hour,
EXTRACT(DAYOFWEEK FROM pickup_datetime AT TIME ZONE "America/New_York") day,
TIMESTAMP_DIFF(dropoff_datetime, pickup_datetime, MINUTE) duration_min,
ST_GEOHASH(ST_GEOPOINT(pickup_longitude, pickup_latitude),6) AS pickup_geohash,
ST_GEOHASH(ST_GEOPOINT(dropoff_longitude, dropoff_latitude),6) AS dropoff_geohash
FROM bigquery-public-data.new_york.tlc_yellow_trips_2015
WHERE (pickup_latitude BETWEEN -90 AND 90)
AND (dropoff_latitude BETWEEN -90 AND 90)
AND (pickup_longitude BETWEEN -180 AND 180)
AND (dropoff_longitude BETWEEN -180 AND 180) )
SELECT AVG(alpha.total_amount) AS avg_cost_no_tolls,
AVG(beta.total_amount) AS avg_cost_with_tolls,
AVG(alpha.duration_min) AS avg_dur_no_tolls,
AVG(beta.duration_min) AS avg_dur_with_tolls
FROM trips as alpha, trips as beta
WHERE alpha.pickup_geohash = beta.pickup_geohash
AND alpha.dropoff_geohash = beta.dropoff_geohash
AND alpha.pickup_geohash = 'dr72wn'
AND alpha.dropoff_geohash = 'dr5rvp'
AND alpha.hour = beta.hour
AND alpha.day = beta.day
AND alpha.hour = 13 --1pm
AND alpha.day = 1 --Sunday
AND alpha.tolls_amount = 0
AND beta.tolls_amount > 0

```

Figure 3: Q2: Compare cost and duration for a given hour and day from 'dr72wn' to 'dr5rvp' with and without tolls, with none of the constraints in Table 1 enabled.

Row	avg_cost_no_tolls	avg_cost_with_tolls	avg_dur_no_tolls	avg_dur_with_tolls
1	32.8	45.41	25.0	27.0

Figure 4: Q2: Compare cost and duration for a given hour and day from 'dr72wn' to 'dr5rvp' with and without tolls -the query answer set is the same for all approaches.

```

WITH trips AS (SELECT *,
EXTRACT(HOUR FROM pickup_datetime AT TIME ZONE "America/New_York") hour,
EXTRACT(DAYOFWEEK FROM pickup_datetime AT TIME ZONE "America/New_York") day,
TIMESTAMP_DIFF(dropoff_datetime, pickup_datetime, MINUTE) duration_min,
ST_GEOHASH(ST_GEOPOINT(pickup_longitude, pickup_latitude),6) AS pickup_geohash,
ST_GEOHASH(ST_GEOPOINT(dropoff_longitude, dropoff_latitude),6) AS dropoff_geohash
FROM bigquery-public-data.new_york.tlc_yellow_trips_2015
WHERE (pickup_latitude BETWEEN -90 AND 90)
AND (dropoff_latitude BETWEEN -90 AND 90)
AND (pickup_longitude BETWEEN -180 AND 180)
AND (dropoff_longitude BETWEEN -180 AND 180) )
SELECT MIN(total_amount) min_cost, MAX(total_amount) max_cost,
AVG(total_amount) avg_cost, MIN(duration_min) min_time,
MAX(duration_min) max_time, AVG(duration_min) avg_time,
count(*) count_trips
FROM trips
WHERE day = 3 --Tuesday
AND hour = 3 --3am
AND pickup_geohash = 'dr5ru7'
AND dropoff_geohash = 'dr5rue'

```

Figure 5: Q3: AVG, MIN, MAX cost and duration for given hour and day from 'dr5ru7' to 'dr5rue', with none of the constraints in Table 1 enabled.

Row	min_cost	max_cost	avg_cost	min_time	max_time	avg_time	count_trips
1	3.3	42.85	8.685347222222221	1	1438	8.422480620155042	6192

(a) Q3 Answer set for lazy cleansing.

Row	min_cost	max_cost	avg_cost	min_time	max_time	avg_time	count_trips
1	3.3	42.85	8.685347222222218	1	1438	8.422480620155037	6192

(b) Q3 Answer set for custom cleansing.

Row	min_cost	max_cost	avg_cost	min_time	max_time	avg_time	count_trips
1	4.3	42.85	8.739071440808633	1	1438	8.542744560561932	5837

(c) Q3 Answer set for eager cleansing.

Figure 6: Q3: AVG, MIN, MAX cost and duration for given hour and day from 'dr5ru7' to 'dr5rue': required cleansing vs eager cleansing, for respectively lazy, custom and eager virtual cleansing.

Measuring Quality of Workers by Goodness-of-Fit of Machine Learning Model in Crowdsourcing

Yu Suzuki
Gifu University
Gifu, Japan
ysuzuki@gifu-u.ac.jp

ABSTRACT

In this paper, we propose a method for predicting the quality of crowdsourcing workers using the goodness-of-fit (GoF) of machine learning models. We assume a relationship between the quality of workers and the quality of machine-learning models using the outcomes of the workers as training data. This assumption means that if worker quality is high, a machine-learning classifier constructed using the worker's outcomes can easily predict the outcomes of the worker. If this assumption is confirmed, we can measure the worker quality without using the correct answer sets, and then the requesters can reduce the time and effort. However, if the outcomes by workers are low quality, the input tweet does not correspond to the outcomes. Therefore, if we construct a tweet classifier using input tweets and the classified results by the worker, the prediction of the outcomes by the classifier and that by the workers should differ. We assume that the GoF scores, such as accuracy and F1 scores of the test set using this classifier, correlates to worker quality. Therefore, we can predict worker quality using the GoF scores. In our experiment, we did the tweet classification task using crowdsourcing. We confirmed that the GoF scores and the quality of workers correlate. These results show that we can predict the quality of workers using the GoF scores.

CCS CONCEPTS

• **Human-centered computing** → *Empirical studies in collaborative and social computing; Computer supported cooperative work;*

KEYWORDS

Crowdsourcing, Machine Learning

ACM Reference Format:

Yu Suzuki. 2021. Measuring Quality of Workers by Goodness-of-Fit of Machine Learning Model in Crowdsourcing. In *25th International Database Engineering & Applications Symposium (IDEAS 2021)*, July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 7 pages. <https://doi.org/10.1145/3472163.3472279>

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 978-1-4503-8991-4/21/07...\$15.00

<https://doi.org/10.1145/3472163.3472279>

1 INTRODUCTION

Crowdsourcing is widely used in many situations, such as constructing training data for machine learning and information systems evaluations. Requesters give HITs (Human Intelligent Tasks) to workers, and the workers complete the HITs. Quality control is one main issue for crowdsourcing because not all workers process tasks honestly.

Quality of outcomes by crowdsourcing is one of the most critical issues because many low-quality workers called *spammers*, who always do wrong behaviors [1]. The spammers aim to earn wages without effort. Therefore, some spammers do not read the tweets and randomly select one label for all tweets. Many researchers propose methods in order to avoid reducing the quality of outcomes by spammers.

Majority voting (MV) is one simple and effective algorithm against spammers. The requesters assign more than two workers to one tweet. Then, the requester selects the most voted label. Therefore, if the labels selected by the spam workers do not win a majority, these labels are not selected. In this way, we can keep the quality of outcomes.

However, the quality of the outcomes is low if there are so many spammers as the workers. Several algorithms are proposed based on MV and EM algorithms to improve the quality of outcomes, such as Dawid & Skene [2], GLAD [3], and a system by Raykar et al. [4]. If the number of workers assigned for each tweet is sufficient and averaging quality of workers is relatively high, the consensus of the voting is correct. However, we cannot calculate the aggregation result in real-time because the outcomes by the other workers are required to calculate majority votes. Moreover, the monetary cost increases if we assign many workers to each tweet.

One feature of our approach is that we can identify spammers without using the outcomes of the other workers. Here we define worker quality. Worker quality value is the number of the correct votes to the number of all votes. We used majority voting for deciding these correct votes. However, using existing methods to calculate worker quality is time-consuming because we can calculate the worker quality values after processing all HITs. We cannot calculate the worker quality values during processing HITs because the outcomes of all HITs are required to process. Therefore, we need an algorithm for predicting these worker quality values without using the outcomes of all workers.

Our approach is to identify spammers without using the outcomes of the other workers. We deal with the goodness-of-fit (GoF) of machine learning models for predicting the quality of workers. We used the accuracy score and F1-score of test data, which indicates the generation capability, as a goodness-of-fit. If workers

appropriately label tweets, the outcomes of workers can be accurately predicted by machine learning models that have sufficient expressive power because the spam workers who randomly select labels to the tweets do not correlate to the tweets. Therefore, if we use these tweets and the labels as a training data set to construct a machine learning-based classifier, the accuracy of a test data set should be low. On the other hand, the input tweets and output labels of high-quality workers should correlate with each other, the classifiers using the input tweets and output labels as training data should be high accuracy.

In this paper, we consider multiple classification tasks of tweets. For example, if a requester needs to collect tweets related to COVID-19 using machine learning, the requester needs training data. Therefore, the requester constructs a task such that the workers select whether the tweets are related to COVID-19 or not. When the workers receive a tweet from a crowdsourcing platform, the workers submit a label from labels such as “related.” and “unrelated.” This paper considers multi-label text classification crowdsourcing tasks.

We conduct experiments to confirm a correlation between the quality of workers and the value of goodness-of-fit of the machine learning model. In our experiment, we did a tweet classification task related to COVID-19 using crowdsourcing. We hired 151 workers and votes whether the tweets are related to COVID-19 or not. Using the tweets and the votings, we generate a classifier using BERT (Bidirectional Encoder Representations from Transformers) [5], a state-of-the-art machine learning model which can be used as a classifier. We did 10-fold cross-validation for each worker and calculated the accuracy and F1-score as goodness-of-fit (GoF) scores. We also calculate an accuracy rate using majority voting and calculate the correlation between the GoF scores and the acceptance rate.

The contributions of this paper are as follows:

- We confirm that the values of goodness-of-fit for the machine learning model and the quality of the workers correlate to each other.
- Using the goodness-of-fit scores, the systems can measure the quality of the workers.
- More than 200 HITs for each worker are required to calculate accurate worker quality

2 RELATED WORK

As we described in Section 1, majority voting-based aggregation methods are proposed for improving the quality of outcomes. However, when we use these methods for improving the quality of outputs, we should assign more than two workers to each HIT. Okubo et al. [6] discovered that more than four workers are required for binary decisions accurately if there are more than half of the workers who always have correct answers. In these methods, the requesters should pay wages to at least four workers for each HIT; the wages of this task increase to keep the quality of outputs. Many researchers proposed methods for measuring worker quality in order to decrease wages.

One possible solution is to capture workers’ behaviors for predicting the quality of workers [7, 8]. Moayedikia et al. [9] proposed a method using EM algorithm. Rzeszotarski et al. [10] proposed a method for predicting worker quality from workers’ behaviors, such as task time and mouse movement, which can be observed

using Javascript. We [11] proposed capturing more behaviors by adding the other operations to workers. For example, when workers classify tweets, the workers should browse the tweets. We added a button to the work screen. The workers browse the tweets during the time the workers are pressing the button.

The purpose of our method is to measure worker’s quality without using the other workers’ outcomes. The purpose of our research is similar to the research by Rzeszotarski et al. However, these existing methods only consider workers’ behaviors, but they do not consider the workers’ outcomes. We believe that the outcomes of the workers should be treated as a behavior of the workers. To our best knowledge, this is the first time to predict worker quality using the outcomes of each worker. In our method, we try to capture features from the outcomes using machine learning models and confirm whether the features correlate the worker quality. Combining these methods with our proposed method, we will measure the worker quality accurately.

3 METHOD

In this section, we describe how to calculate GoF scores for each worker. We measure the worker quality by the following three steps:

- (1) Process HITs by workers
- (2) Construct machine learning model using the worker’s outputs
- (3) Measure the value of goodness-to-fit by 10-fold cross-validation

We deal with majority voting-based worker evaluation systems, which we call a point system to increase outcomes. This system is based on majority voting. The system assigns the tweets to multiple workers and aggregates outcomes of the workers using majority voting. If the label the worker select is a majority, the worker wins more than 1 points, but if the label is not a majority, the worker wins only 0.01 points. We explain this algorithm in Section 3.3.

3.1 Process tasks by workers

We prepare a set of tweets $T = \{t_1, t_2, \dots, t_N\}$ using keywords. Our task aims to collect tweets related to COVID-19; we set keywords as “COVID” and “Corona” for filtering tweets.

We hired a set of workers $W = \{w_1, w_2, \dots, w_M\}$ using a crowdsourcing platform. We did not select workers, and we did not measure the qualities of workers using pre-tasks before processing our task. All workers who apply this task do our tasks.

We prepared the following labels $L = \{l_0, l_1, l_2, \dots, l_6\}$ as follows:

- l_0 : The tweet is related to COVID-19.
 - l_1 : The tweet includes only a fact which is publically available.
 - l_2 : The tweet includes only a fact which is not publically available.
 - l_3 : The tweet includes optionions of a twitter user who posts this tweet.
 - l_4 : I (the worker) cannot understand whether the tweet includes facts and opnions or not.
- l_5 : The tweet is not related to COVID-19.
- l_6 : I (the worker) cannot select from the labels l_0, l_1, \dots, l_5 .

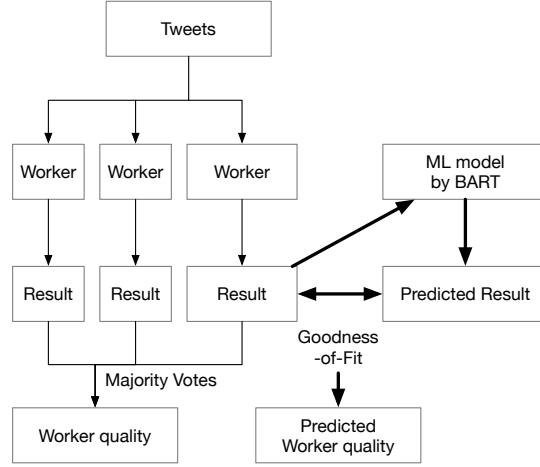


Figure 1: Overview of our system. Majority vote part is required for evaluating our proposed system, not required for production.

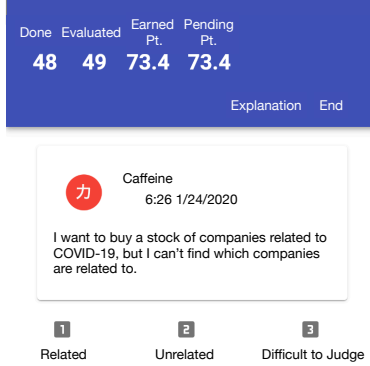


Figure 2: Screen capture of labeling tweets. Original work screen is written in Japanese. All descriptions are translated for convenience.

When the workers select l_0 , the workers select one labels from l_1 , l_2 , l_3 , and l_4 . Therefore, l_0 is not used for labeling tweets.

For example, if there is a tweet such that “New York city is locked down,” the workers should select l_1 . However, if there is a tweet “New York city is locked down. I wanna go shopping but...,” the workers should select l_3 , not l_1 because this tweet includes both a fact and an unclear opinion. The workers should select l_2 for a tweet “My father gets coronavirus twice.” We explained this policy to the workers with many examples before labeling tweets.

We show an instruction of this task to all workers. This instruction includes explaining labels with some examples and the point system described in Section 3.2. We did not explain how to calculate the points.

Fig. 2 shows a screen capture image of labeling application for workers. There is a tweet for labeling at the center of this screen. The workers browse this tweet, then the workers first select one of three labels l_0 , l_5 , and l_6 . If the workers select l_0 , the workers select one of four labels l_1 , l_2 , l_3 , and l_4 . A set of outcomes $V(w_i)$

by worker w_i is as follows:

$$V(w_i) = \{(t_a, l(t_a, w_i)), (t_b, l(t_b, w_i)), \dots, (t_{k(w_i)}, l(t_{k(w_i)}, w_i))\} \quad (1)$$

where $t_a, t_b, \dots, t_{k(w_i)} \in T$ are tweets which w_i processes, $k(w_i)$ is a number of tweets which w_i processes, $l(t_a, w_i) \in L$ is a label for t_a by w_i . If w_i process one tweet, the number of “Done” at the left top of Fig. 2 increments.

The system should not pay wages to spammers in order to increase the quality of outcomes. However, if there are spammers, they process many tweets in a short time and earn wages a lot. To avoid this problem, we construct a point system that continuously measures the quality of workers. In the next section, we explain this system in detail.

3.2 Earn Points

To avoid spamming, we deal with the majority voting-based point system. In this system, the workers win points if the labels the workers select are the majority. To earn wages, workers must earn points. Therefore, if the workers are spammers, the points increase slowly, then they do not earn enough wages.

Our system assigns five workers to each tweet. Then, the system receives five labels for each tweet. If the number of labels by majority voting is less than half the number of all labels, the system assigns one more worker and receives a label. If the number of labels by majority voting is more than half the number of all labels, or if the number of workers is more than ten, the system stops assigning workers.

Then, the system gives the points to the workers:

- If the workers did not select labels by majority voting, the workers get 0.01 points.
- If the workers select labels by majority voting, the workers get p points as follows:

$$p = \frac{|l_m|}{\sum^i |l_i|} \quad (2)$$

where l_m is a majority label for the tweet, and $|l_m|$ is a number of workers who select l_m . $\sum^i |l_i|$ is a number of workers who is assigned to the tweet. For example, if first five workers select the same label, the number of $|l_m|$ and $\sum^i |l_i|$ are the same, then the workers earn 1 point.

The point is calculated when the system stops assigning workers to the tweet. That is, the point is not calculated immediately after labeling the tweet by the worker. The number appeared at the left top of the work screen shown in Fig. 2. “Done” means the number of HITs that the worker process, “Evaluated” means the number of tweets in which the point is calculated. The following number, “Earned Pt.” means a total point the worker earns. The number “Pending Pt.” means the number of points that are not converted to the wages. If the number of *Earned Point* is more than a threshold which the requesters decide, the workers earn wages.

Our system does not inform whether the labels which the workers select are the majority or not.

3.3 Measure GoF scores

GoF scores are the averaging scores of accuracy scores or F1 scores of the test set, which means how much the machine learning-based classifier fits the outcomes of the workers. We calculate the GoF scores for each worker using stratified 10-fold cross-validation because the number of tweets for each label may become imbalanced. We construct a machine learning-based classifier which imitates the votes of the worker. We used BERT (Bidirectional Encoder Representations from Transformers) [5], one state-of-the-art text classifier, for building the classifier.

We randomly divide $V(w_i)$ into 10 sets $V_1(w_i), V_2(w_i), \dots, V_{10}(w_i)$. First, we set $V_1(w_i)$ as a test set and the other sets as a training set. We construct a machine learning model and measure the accuracy a_1 and F1-score f_1 using a test set. Next, we set $V_2(w_i)$ as a test set and do the same process. We do this process 10 times and calculate a_1, a_2, \dots, a_{10} and f_1, f_2, \dots, f_{10} .

Finally, we calculate the averaging score of accuracy $A(w_i)$ and that of f1-score $F(w_i)$. These values show the goodness-of-fit of the machine learning model to the worker w_i ’s outcomes. We used $A(w_i)$ or $F(w_i)$ as a quality value of workers.

4 EXPERIMENTAL EVALUATION

We did the experiments for two purposes:

- (1) Is it possible to measure the quality of workers using the values of goodness-of-fit of the machine learning model, such as $A(w_i)$ and $F(w_i)$?
- (2) How many HITs should the worker process for predicting the accurate quality of workers?

We performed two experiments. Experiment 1 confirms whether $A(w_i)$ or $F(w_i)$ correlate to the worker qualities. Experiment 2 is to discover how many HITs the workers should process for calculating accurate GoF scores.

4.1 Experimental Setup

We collect 51,218 tweets related to COVID-19 using Twitter Sampled stream v1 API¹ with keywords “COVID” and “Corona.” All of

¹<https://developer.twitter.com/en/docs/labs/sampled-stream/>

Table 1: The number of tweets by majority voting and the number of votes

label	# tweets	# votes
l_1	12,148	72,917
l_2	4,829	56,605
l_3	31,819	162,635
l_4	62	4,047
l_5	4,754	55,872
l_6	27	3,019
all	50,696	355,095

the tweets are posted between Jan. to May 2020 and written in Japanese. These tweets do not include retweets and replies. Tweets which have URL are also removed. We did not select tweets manually, then some tweets are not related to COVID-19 but include the string Corona or COVID. such as the tweets related to Corona beer or the tweets related to a video streaming service named niconicovideo because the string “niconicovideo” includes the string “COVID.”

We constructed a tweet labeling system. We used Ruby on Rails 6.0.3² and PostgreSQL 13³ as a backend system, and React⁴ as a frontend system. This frontend system is suitable for smartphones; more than half of the workers use smartphones to process the task.

We hired 195 workers from CrowdWorks⁵, a major crowdsourcing platform in Japan. The system assigns between five and ten workers. In our system, the workers earn wages 200JPY \approx 2USD for every 500 points \approx 500 HITs.

There is no gold data, which means there are no known priory answers. Therefore, we generate gold data using majority voting. Table 1 shows the number of tweets by majority votings. There are 2,943 tweets with more than two majority labels; we count the tweets at each label. The number of tweets labeled l_4 , and l_6 means the workers cannot select the labels. Because the workers can decide labels for almost all tweets. Therefore, we ignore tweets labeled l_4 and l_6 in our experiments. We constructed four-label (l_1, l_2, l_3 , and l_5) classifiers for each worker to calculate the GoF scores. Then, we measure acceptance rate $R(w_i)$ which we consider worker quality as a gold data for each worker as follows:

$$R(w_i) = \frac{|V(w_i) \cap MV|}{|V(w_i)|} \quad (3)$$

where $|V(w_i)|$ is the number of votes by w_i . MV is a correct label by majority votings. $|V(w_i) \cap MV|$ is the number of votes labeled by majority voting, and w_i are the same.

We constructed a machine learning model using BERT. We used pretrained Japanese BERT models and tokenizer⁶. We set the number of epochs to 200 and the number of early stopping to 20.

4.2 Experiment 1: Predicting worker quality using GoF scores

Fig. 3 shows the results of this experiment. Each point corresponds to one worker. A Pearson correlation coefficient of accuracy is 0.23,

²<https://rubyonrails.org>

³<https://www.postgresql.org>

⁴<https://github.com/facebook/react>

⁵<https://crowdworks.jp/>

⁶<https://github.com/cl-tohoku/bert-japanese>

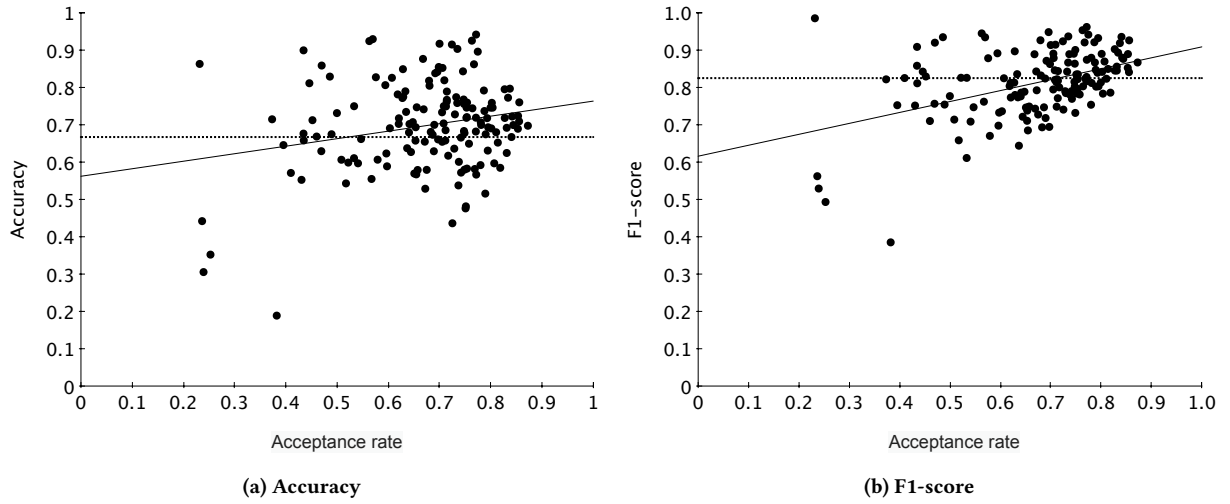


Figure 3: Averaging accuracy score and averaging F1-score vs. acceptance rate. Each line shows a regression line, and each dotted line shows an averaging accuracy score and averaging F1-score of the classifiers with the labels by majority votes as training and test dataset.

and that of the F1-score is 0.444. p -value of accuracy is $4.80 \cdot 10^{-3}$ and that of F1-score is $1.16 \cdot 10^{-8}$, both values are lower than 0.05. From this result, we conclude that both accuracy score and F1 score correlate to the worker quality. Therefore, using either accuracy score or F1 score, we can predict worker quality accurately. Also, the F1 score is better than the accuracy score to predict worker quality values.

We observed the results in detail, and we found that both the accuracy scores and the F1 scores are high if the number of HITs the worker processes is small, even if the workers are low-quality. We confirmed that we could identify low-quality workers if the workers do more than 200 tasks and the outcomes of the workers are different from the other good workers. In this experiment, the acceptance rate of 116 of 151 (77%) workers is more than 0.6, which can be considered good workers.

We draw a regression line for each graph. In the graph of accuracy score, the gradients and the intercepts of the regression line are 0.20 and 0.56, respectively. RMSE is 0.12. In the F1-score, the gradients and the intercepts are 0.29 and 0.62, respectively. RMSE is 0.14. From this result, we also confirmed that the F1-score is better than the accuracy score because the gradients of the F1-score are higher than that of the accuracy score, then we can identify high-quality and low-quality workers using the F1 score.

In this experiment, we also construct a classifier that can output labels by majority voting. We prepared input texts, and the labels correspond to each text by majority voting. Then, we calculate the accuracy scores and the F1 scores by 10-fold cross-validation. We show the accuracy score and F1-score as dot lines in each graph of Fig. 3. The accuracy score is 0.668, and the F1-score is 0.826. From this result, we confirmed that predicting labels by majority voting is relatively tricky than predicting labels by each worker. Also, the accuracy of the classifiers which output accurate labels is impractical.

Table 2: Workers' accuracy

Worker	Category	# votes	$R(w_i)$	$A(w_i)$	$F(w_i)$
w_1	TP	4,128	0.787	0.793	0.922
w_2	TN	3,262	0.253	0.101	0.227
w_3	FP	5,809	0.500	0.733	0.778
w_4	FN	830	0.804	0.742	0.785

4.3 Experiment 2: A number of votes for accurate prediction of worker quality

In this experiment, we discover how many votes we need to predict accurate worker quality. We pick up four typical workers from the following four categories: TP) Both $F(w_i)$ and acceptance rate $R(w_i)$ are high, TN) Both $A(w_i)$ and $F(w_i)$ are low, FP) $F(w_i)$ is high and $R(w_i)$ is low, FN) $F(w_i)$ is high and $R(w_i)$ is low. Then, we pick up 50, 100, 150, \dots votes; then we perform 10-fold cross-validation, then we calculate accuracy scores and F1-scores as GoF scores. As shown in Fig. 3b, no worker is in the group FN. Therefore, we choose the worker who has the lowest acceptance rate with the highest F1-score. Table 2 shows the acceptance rate, accuracy, and F1-score of the workers we choose.

Fig. 4 shows time-series variation of the accuracies and F1-scores of each worker. From these figures, we can observe that the accuracy score and F1 scores of all workers are high if the number of votes is small. However, when the number of votes is more than 200, the scores of low-quality workers become low. Therefore, the workers should do more than 200 HITs to calculate accurate GoF scores.

However, our proposed system incorrectly identifies several low-quality workers as high-quality, like worker w_3 . These workers are shown at the left-top part of the graph in Fig. 3. The number of workers in this field is relatively small. We check the outcomes of these workers in detail and found that these workers do HITs in good faith but misunderstand the instructions of this task. In our

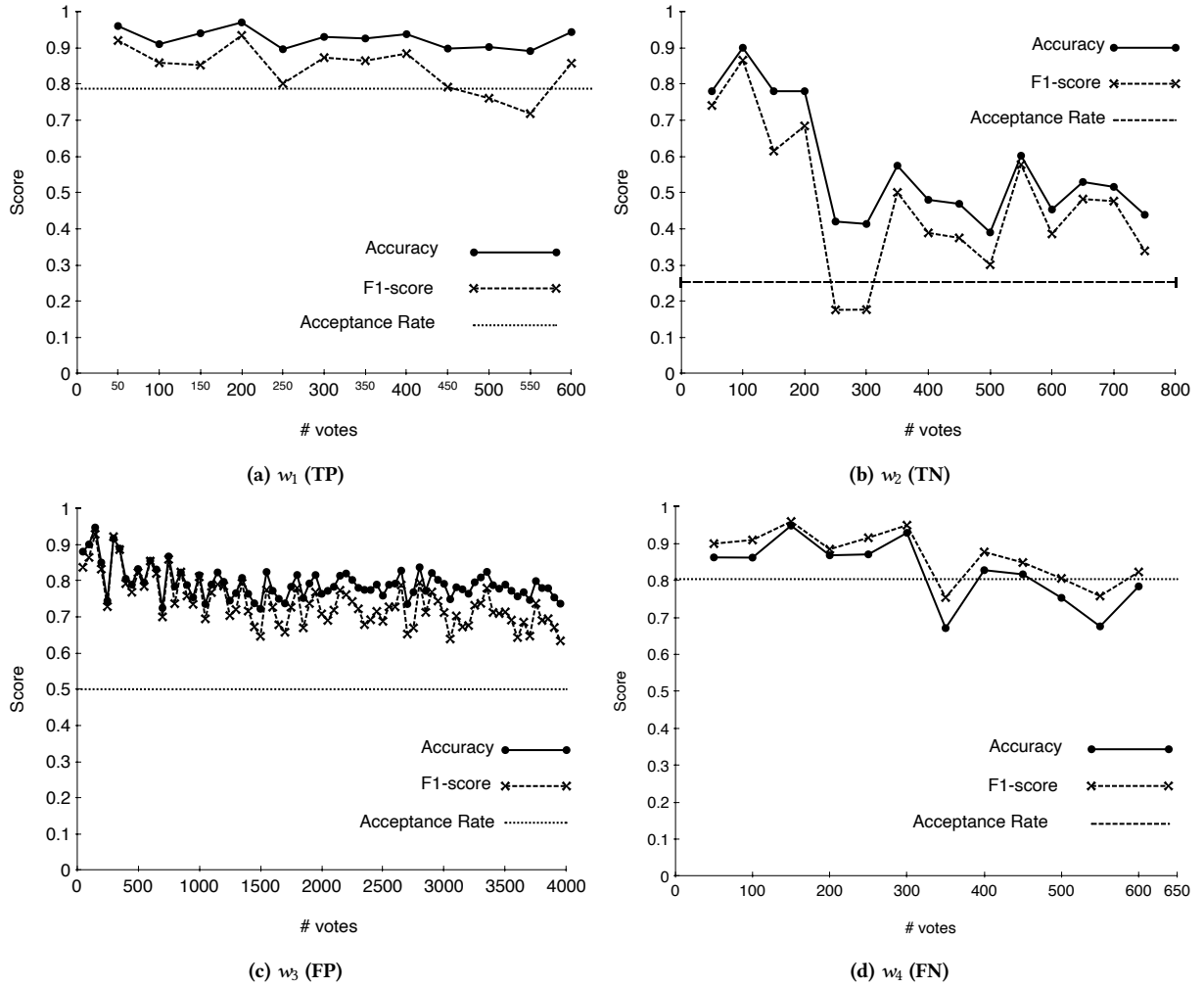


Figure 4: Time series validation of Number of votes vs. Accuracy and F1-score

task, as we described in Section 3.1, if there is a tweet that includes both a fact and an opinion, the workers should select l_3 : the tweet includes opinions of a Twitter user who posts this tweet. However, these workers select l_1 : the tweet includes only a fact. Therefore, these workers should be considered as low. The correspondences of requesters for these workers and that of the workers randomly select the options should differ. We should construct a system for classifying these different types of low-quality workers.

5 CONCLUSION

In this paper, we proposed a method for predicting the quality of workers using the GoF (goodness-of-fit) scores of machine learning models. We consider accuracy scores and F1 scores as the goodness-of-fit scores. We discover whether there is a correlation between the goodness-of-fit scores and the acceptance rate of the workers. We did the crowdsourcing task and measured the goodness-of-fit scores.

We performed an empirical evaluation on detecting high-quality and low-quality workers in a Twitter annotation task. As a result of experiment 1., we found that both the accuracy score and F1 score are correlated to the worker quality, and the F1 score is better than the accuracy score for measuring worker quality. From experiment 2., we found that at least 200 votes are required to calculate accurate GoF scores.

In future work, we plan to integrate our proposed worker quality measurement system into a crowdsourcing platform. Unlike the acceptance rate, the platform can calculate F1 scores in a short time. Therefore, the platform can quickly kick out the low-quality workers, and the requesters save wages and increase the quality of outcomes.

Moreover, we should integrate our method with the other worker quality prediction methods based on worker behaviors, introduced in Section 2. Our method and these methods have a complementary

relationship. In our method, several low-quality workers are classified into high-quality workers. When we combine several methods, the accuracy of this classification will increase.

In our experiment, we deal with the worker evaluation system based on majority voting. This system is used because we do not understand how the GoF scores are helpful. We can develop another point system using GoF scores. However, this system can be a blackbox; the requesters cannot understand why the workers are considered as low-quality even if the workers' outcomes are high quality. To solve these problems, we should develop a method to understand this blackbox.

Acknowledgment. : This work was partly supported by JSPS KAKENHI Great Number 19H04218, 18H03342, and 19H04221.

REFERENCES

- [1] Victor S. Sheng, Foster Provost, and Panagiotis G. Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '08*, pages 614–622, New York, NY, USA, 2008. Association for Computing Machinery.
- [2] A. P. Dawid and A. M. Skene. Maximum likelihood estimation of observer error-rates using the em algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):20–28, 1979.
- [3] Jacob Whitehill, Ting fan Wu, Jacob Bergsma, Javier R. Movellan, and Paul L. Ruvolo. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 2035–2043. Curran Associates, Inc., 2009.
- [4] Vikas C. Raykar, Shipeng Yu, Linda H. Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. Learning from crowds. *Journal of Machine Learning Research*, 11(43):1297–1322, 2010.
- [5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.
- [6] Yuki Okubo, Teruaki Kitasuka, and Masayoshi Aritsugi. A preliminary study of the number of votes under majority rule in crowdsourcing. *Procedia Computer Science*, 22:537 – 543, 2013. 17th International Conference in Knowledge Based and Intelligent Information and Engineering Systems - KES2013.
- [7] Gabriella Kazai and Imed Zitouni. Quality management in crowdsourcing using gold judges behavior. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining, WSDM '16*, pages 267–276, New York, NY, USA, 2016. Association for Computing Machinery.
- [8] Matthias Hirth, Sven Scheuring, Tobias Hofffeld, Christian Schwartz, and Phuoc Tran-Gia. Predicting result quality in crowdsourcing using application layer monitoring. In *2014 IEEE Fifth International Conference on Communications and Electronics (ICCE)*, pages 510–515, 2014.
- [9] Alireza Moayedikia, Hadi Ghaderi, and William Yeoh. Optimizing microtask assignment on crowdsourcing platforms using markov chain monte carlo. *Decis. Support Syst.*, 139:113404, 2020.
- [10] Jeffrey M. Rzeszutarski and Aniket Kittur. Instrumenting the crowd: Using implicit behavioral measures to predict task performance. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology, UIST '11*, pages 13–22, New York, NY, USA, 2011. Association for Computing Machinery.
- [11] Yu Suzuki, Yoshitaka Matsuda, and Satoshi Nakamura. Additional operations of simple hits on microtask crowdsourcing for worker quality prediction. *Journal of Information Processing*, 27:51–60, 2019.

SSstory: 3D data storytelling based on SuperSQL and Unity

Jingrui Li
Dept. of Information and Computer
Science,
Keio University
Yokohama, Japan
li@db.ics.keio.ac.jp

Kento Goto
Dept. of Information and Computer
Science,
Keio University
Yokohama, Japan
goto@db.ics.keio.ac.jp

Motomichi Toyama
Dept. of Information and Computer
Science,
Keio University
Yokohama, Japan
toyama@ics.keio.ac.jp

ABSTRACT

SuperSQL is an extended SQL language, which brings out a rich layout presentation of a relational database with a particular query. This paper proposes SSstory, a storytelling system in a 3D data space created by a relational database. SSstory uses SuperSQL and Unity to generate a data video and add cinematic directions to the data video. Without learning special authoring tooling, users can easily create data videos with a small quantity of code.

CCS CONCEPTS

• **Human-centered computing** → *Visualization systems and tools*.

KEYWORDS

SuperSQL, databases, data visualization, data storytelling

ACM Reference Format:

Jingrui Li, Kento Goto, and Motomichi Toyama. 2021. SSstory: 3D data storytelling based on SuperSQL and Unity. In *25th International Database Engineering & Applications Symposium (IDEAS 2021)*, July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 11 pages. <https://doi.org/10.1145/3472163.3472277>

1 INTRODUCTION

Nowadays, with a mass of data created, how to tell the insights of data to the audience is becoming critical. Therefore the data storytelling approach to telling the data insight as a story becomes a topic in data visualization.

Previous research[1] named SSQ4.5DVS uses Unity[2], a game development platform, to implement a system that can generate objects with data in a 3D space. With SSQ4.5DVS, even users without knowledge of Unity can generate visualized scenes in the 3D space.

However, SSQ4.5DVS is just a visualization tool which not support the function of data storytelling. Also, the authoring tool for the data storytelling existing but such tool takes time for learning or needs specialized knowledge and technology.

This paper proposes data storytelling systems that use SuperSQL, an extended SQL language, and Unity to make a data video in the 3D space.

2 RELATED WORK

Several papers inspired our paper. Edward et al.[3] looks at the construction of data storytelling and analyzes the element. This paper refers to the structure of data storytelling. Fereshteh Amini et al.[4] analyze 50 of professionally designed data videos, extracting and exposing their most salient constituents. Also, Fereshteh Amini et al.[5] develops a tool named DataClips which can generate data video with the data clips. Users can easily do it without using a video authoring tool like Adobe AfterEffects. Ren[8] looks at the importance of annotation in data storytelling and develops a specialized data storytelling tool for the annotation. Edward et al.[6] research the element of stories then sum up the design pattern of the data story with the result. Lyu et al.[7] shows the communication model of storytelling and introduces a visualization works, a Chinese painting, using the big meteorological data in China. Amin Beheshti et al.[9] combines the intelligent DataLake technique and data analytics algorithm to implement an interactive storytelling dashboard for social data understanding. Shi et al.[10] develop a tool that automatically generates data stories from Google Spreadsheet.

However, these related works expend time costs to generate data stories. Also, most of them present the data story as a 2D graph; it is hardly any approach for data storytelling in 3D space.

3 SSstory

This paper proposes an approach that combines the StoryGenerator and the StoryEditor. StoryGenerator uses SuperSQL and Unity to visualize the data from the database, and the StoryEditor adds the story elements to the work of visualization. The presentation of the data story will be a data video, which means show the data as a video or animation.

3.1 High-dimensional visualization object

A High-dimensional visualization object can show high-dimension information. For example, an object like a human face can allocate the height of the nose, the width of the mouth, the color of the eye to high-dimension information.

We can create a High-dimensional visualization world with several High-dimensional visualization objects. The advantage of the High-dimensional visualization object and High-dimensional visualization is that the audiences can choose the information they want to know.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada
© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-8991-4/21/07...\$15.00
<https://doi.org/10.1145/3472163.3472277>

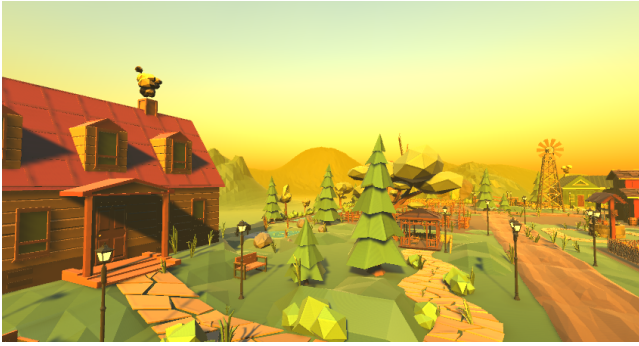


Figure 1: Asset of Unity

Figure 1 shows a result that uses the asset of Unity to create a world just like a farm[14]. In this world, the strength of wind, the strength of sunlight, the number of animals, the amount of grass can have information from the database.

3.2 About SuperSQL

SuperSQL is an extended language of SQL that structures the output result of the relational database, enables various layout expressions, and has been developed at Keio University Toyama Laboratory[11][12]. The query replaces the SQL SELECT clause with a GENERATE clause with the syntax of GENER-ATE <media> <TFE>, where <media> is the output medium and can specify HTML, PDF, etc. Also, <TFE> represents a Target Form Expression that is an extension of the target list and is a kind of expression having a layout specifying operators such as a connector and an iterator.

3.3 Architecture

Figure 2 shows the architecture of the SSstory. The system consists of the StoryGenerator and the StoryEditor. The StoryGenerator consists of the Parser that distinguishes the SQL query and layout presentation, the DataConstructor that layout the data from the result of the database and structured information of the table, the CodeGenerator to generate XML file according to the structured data. In the StoryEditor, the Timeline and Cinemachine package is used to improve the quality of DataVideo. This system aims to generate 80% of the material by StoryGenerator and finish the video with the rest 20% adjustment by StoryEditor.

Then we will show the workflow of this system. First, store the data user wants to visualize in the database. When the SuperSQL query is executed, the XML file and C# file are generated based on the data and the query contents. The information of the object and the elements given on the object and the layout are described in the XML file. C# file reads its XML file and creates objects, gives color and animation for objects, and determines the position of each object from layout information. By importing these two script files into Unity and executing it, we can realize the data visualization specified in the query. In the case of using assets, we also need to import assets to Unity at the same time. Then in the StoryEditor, the Cinemachine and AnimatorController will create the animation element through the XML file. Then the user can use the Timeline to

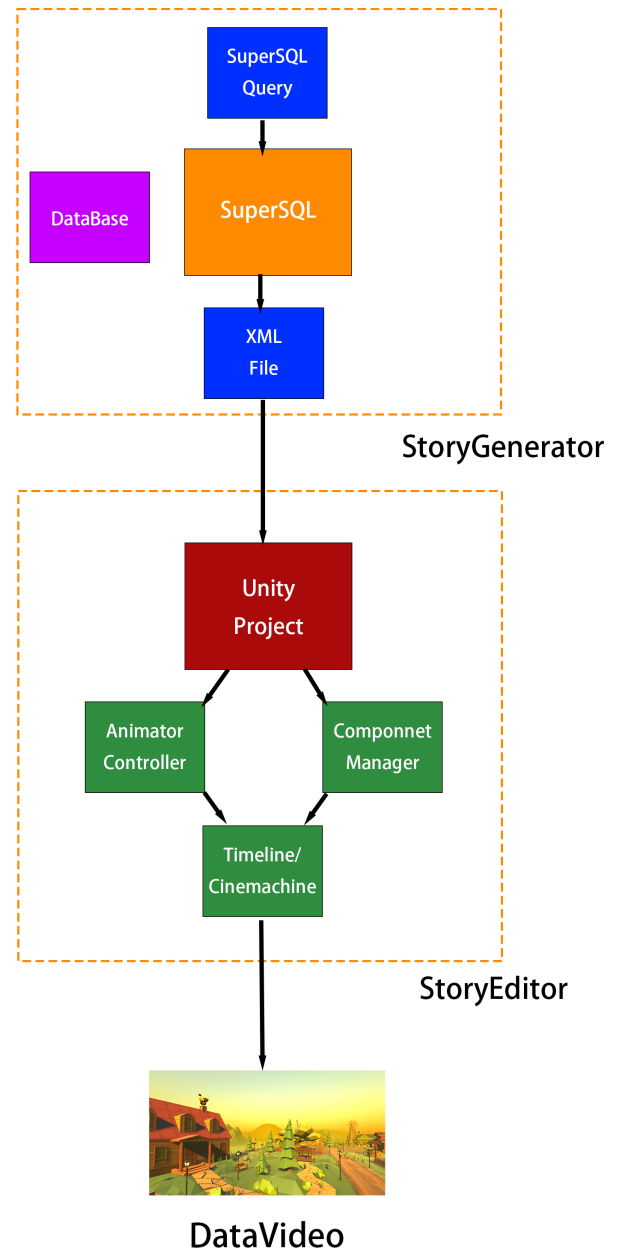


Figure 2: Architecture of SuperSQL

choose the activated camera and adjust the video sequence. Figure3 shows the Timeline used by StoryEditor.

3.4 Object Generation Function

This system uses the asset function to create objects which property likes size can be assigned with the argument. The system supports three types of object generation functions.

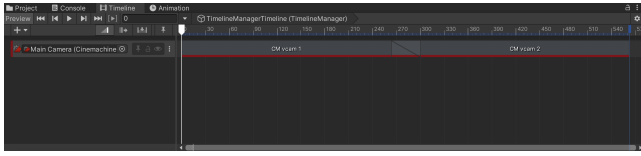


Figure 3: Timeline of Unity

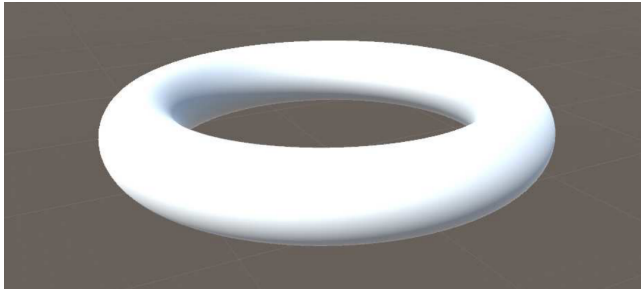


Figure 4: Generation example of object(torus)

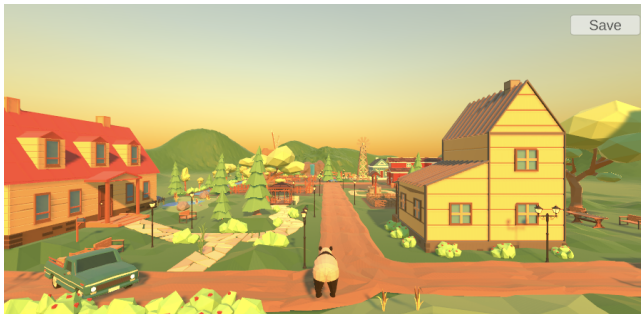


Figure 5: Generation example of asset(panda)

3.4.1 Primitive Generation Function. User can create primitive object like cube, sphere, torus, cuboid, pyramid by using this function.

object(object, argument1, argument2)

The following shows a description example of the object function for generating an object (torus) and its generation result.

object('torus', r1, r2)

Also the 3D text object could be generated though the primitive generation function

object('text', text, size)

3.4.2 Asset Generation Function. Asset generation function uses the asset, an existed object created by the user.

asset(asset_name, size)

Figure5 user can generate an asset called Pandan through the following description.

asset('Panda', size)

3.4.3 Random Generation Function. The random generation function will generate objects from a folder. All the assets in the folder will be put in an array and wait to be generated. By default, assets will be generated in the X-Y plane with normal distribution, and the max coordinate is according to the scene asset.

random(asset_folder, number, min_size, max_size)

For example, this query can generate all assets in the folder of Animal according to the argument. If the user wants to generate them in the whole X-Y-Z space, not the X-Y plane, the user can write a decoration(@) for the function.

random('Animal', number, min_size, max_size)@{space = 'X-Y-Z'}

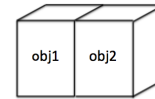
3.5 Connector

Table 1 shows the operator supported by the SuperSQL. The connector is an operator that specifies in which direction (dimension) the data obtained from the database is to be combined. There are the following three kinds. The character used to represent each connector in a SuperSQL query is noted in parentheses.

- Horizontal Connector(,).

Two pieces of data connected by this connector are arranged next to each other horizontally (along with the x-axis). In a flat document, this means the two pieces of data are shown on the same line. In a 3D document, this connector acts as shown below.

ex: obj1, obj2



- Vertical Connector(!).

Two pieces of data connected by this connector are arranged next to each other vertically (along the y-axis). In a flat document, this means the two pieces of data are shown on two separate lines. In a 3D document, this connector acts as shown below.

ex: obj1! obj2



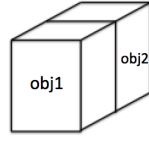
- Depth Connector(%).

Two pieces of data connected by this connector are arranged next to each other in-depth (along with the z-axis). This means the first piece of data links to the second piece of data in a flat document. In a 3D document, this connector acts as shown below.

ex: obj1% obj2

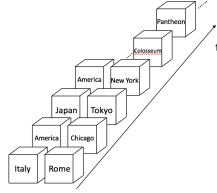
- Time Connector(#).

Two pieces of data connected by this connector are arranged next to each other in the time axis. It will be shown at a constant time interval. Following description shows an example



for Time Connector.

ex : [country, city]##[building]#



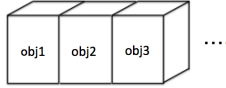
3.6 Grouper

The second type of operator available in SuperSQL is the grouper. When a grouper is used, the attribute or group of attributes affected by the operator is repeated in the document for each tuple retrieved from the database by the query. The following groupers are available in this system.

- Horizontal Grouper([]).

An object or group of objects is added to the document for each tuple in the relation retrieved by the query, each object is arranged horizontally. In our 3D system this produces the pattern shown below.

ex : [Obj],



- Vertical Grouper([]!).

An object or group of objects is added to the document for each tuple in the relation retrieved by the query, each object is arranged vertically. In our 3D system this produces the pattern shown below.

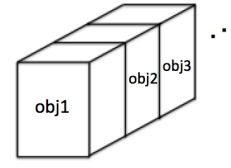
ex : [Obj]!



- Depth Grouper([]%).

An object or group of objects is added to the document for each tuple in the relation retrieved by the query; each object is arranged in depth. In our 3D system, this produces the pattern shown below.

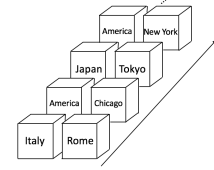
ex : [Obj]%



- Timeline Grouper([]#).

The timeline grouper will arrange scenes to a timeline and present the data as a data video.

ex : [country, city]#



- Slideshow Grouper([]&).

Slideshow grouper will arrange scenes like a slideshow with a UI button. Click the button can translate to another scene. Chapter 4.3 will show a example for slideshow repeat.

Table 1: Operator in SuperSQL

tfe1, tfe2	Horizontal connector
tfe1! tfe2	Vertical connector
tfe1% tfe2	Depth connector
[tfe],	Horizontal grouper
[tfe]!	Vertical grouper
[tfe]%	Depth grouper
[tfe]#	Timeline grouper

3.7 Environmental Function

As a result of the environmental function, the information of the environmental object will be assigned. The environmental object is a kind of object that unique exist in the scene. For example, there is only one sun in the scene, and it will be an environmental object. Through the following query, the user can vary the strength of sunlight along the timeline.

env(object, component, attribute, value)

When the environmental function is executed, an empty GameObject called EnvController will be generated in Unity, and the script of the EnvController will fetch the environment object and its component, assign a value to the attribute.

env('Sunlight', 'Light', 'intensity', value)

This function can assign the value of for the Sunlight object to change the lighting in the scene.



Figure 6: Example of annotation

3.8 2D Element Function

This system supports two types of 2D element functions. 2D element means an object that exists in the canvas. Any connector will not influence its coordinate so that it could be written in any place in the code. The 2D element also could be repeated by a timeline grouper to generate an animation.

3.8.1 Annotation Function. An annotation function will be used to create an annotation for canvas.

```
annotation(type, text)
```

The first argument shows the type of annotation. There are some presets for the annotation like "text_bottom," which means a text box at the bottom of the canvas.

```
annotation('text_bottom', 'some annotations')
```

Figure 6 shows the result of this query. Also, you can use decoration to complete the detail of annotation such as width, height, font size, texture, and highlight.

```
annotation('text_bottom', 'some annotations')@{font_size = '20'}
```

Fig. will show the result of this query and its decoration. Another valuable decoration for the annotation function is setting the start time to achieve a better presentation for the data video.

```
annotation('text_bottom', 'some annotations')@{start_time = '5'}
```

The annotation will appear after 5 seconds, according to the decoration. Fig shows a combination of three annotation functions.

```
[annotation('text_bottom', data)]#
```

The timeline grouper with the annotation function can make the annotation to be an animation annotation to show the data.

3.8.2 Chart Function. Chart function will be used to create charts in the canvas. The first argument shows the type of chart. Some presets of charts have been prepared. The data of the chart is assigned by the second and the third argument.

```
chart(type, data1, data2)
```



Figure 7: Graph And Chart

We use the asset of Unity called 'Graph and Chart'[13] to implement this function. The table shows the supported type of chart in the chart function. Figure 7 shows a sample of Graph And Chart asset.

```
[chart(type, data1, data2)], [chart(type, data1, data2)]!
```

Both horizontal grouper([,]) and vertical grouper([!]) will repeat the data and make it be a whole chart. The graph shows the grouper used for the chart function.

```
[chart(type, data1, data2)]#
```

The timeline grouper can make the chart to be an animation chart.

Like the annotation function, you can use the decoration to adjust the chart, like width, height, axis label, and approximate curve.

```
chart(type, data1, data2)@{approximate_curve = 'linear'}
```

Two charts can be combined as one chart by using the mark of '+',

```
chart('Bar', data1, data2) + chart('LineGraph', data3, data4)
```

3.9 Optional function

The optional function is a function used to add an extra attribute to the object generation function. This section will introduce the optional function and show some examples.

```
function(<TFE>, parameter)
```

The first argument <TFE> means the object generation function with its connector and grouper. The second argument shows the extra attribute given to the TFE.

- Hop Function

The hop function can cause an object to jump by describing velocity and vertex and axis direction. The following shows a description example of hop function and its image.

```
hop(<TFE>, velocity, vertex, axis)
```

- Pulse Function

The pulse function can stretch the object by describing magnification and velocity. The following shows a description example of pulse function and its image.

```
pulse(< TFE >, magnification, velocity)
```

- Rotate Function

The pulse function can rotate the object by describing the rotation rate for three axes. The following shows a description example of pulse function and its image.

```
rotate(< TFE >, X_rate, Y_rate, Z_rate)
```

- Color Function

The color function gives an object the color by describing with a character string. Available colors include "red", "blue", "green", "black", "clear", "cyan", "gray", "yellow", "white", "magenta". The following shows a description example of color function.

```
color(< TFE >, color_name)
```

- Position Function

The position function is a valuable function to change the absolute coordinate for the object. It original point(0, 0, 0) of 3D space as a center and update the coordinate after generation.

```
position(< TFE >, x_coordinate, y_coordinate, z_coordinate)
```

- Move Function

The move function can change the relative coordinate for the object. The move function is different from other functions. If a move function is described in a move function again, the coordinate will be calculated through all the move functions.

```
move(< TFE >, x_coordinate, y_coordinate, z_coordinate)
```

- Optional Annotation Function

The optional annotation function is a function to assign annotations to an object. Unlike the annotation function in the section, which generates annotation in the canvas, the optional annotation function adds some annotations to an object. The user can get the annotation by clicking the object in the game mode.

```
optional_annotation(< TFE >, text)
```

- Camera Function

Camera and cinema direction play an important role in data storytelling. This system supports the camera function to generate a camera in the scene. The type of camera including follow camera, a camera that follows an object, a fixed camera, a camera in a stable position, and FPS camera, a camera to show the first-person view of an object.

```
camera(< TFE >, type)
```

Figure8 shows a follow camera for a deer asset.

```
camera(asset('Deer', 1), 'Follow')
```

Moreover, the camera function can also be repeated by grouper to generator a camera for each object. Decoration also can be used to change the attribute of the camera, such as lens, damping, and coordinate.

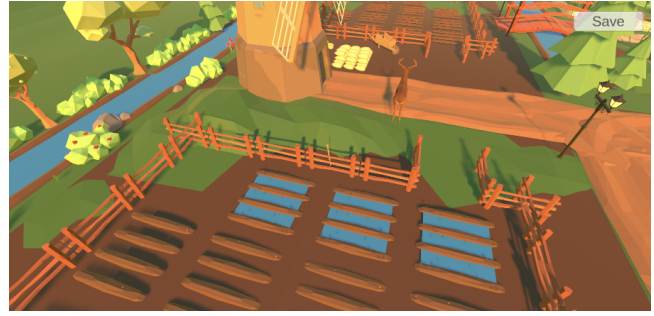


Figure 8: Example of follow camera

3.10 Query

In general, the query of storytelling will be the following expression.

```
GENERATE unity_dv
Scene(scene_name)
[option to object(
several functions(...), necessary
↪ parameter)(grouper)(connector)
(environmental function)
(2D element)
FROM table
WHERE condition
```

(First element)Describe the scene used in this system.

(Second element)SSstory supports several functions to assign the information to objects. Decide which element assign to the objects through the function name, then describe the object generate function or other function in the first argument. It is necessary to describe the argument in the correct order. The connector will be used to connect another object here. Each of the objects will be the same expression rule.

(Third element)Assign information to environmental objects through the environmental function. There is only one environmental function for an environmental object in a scene.

(Fourth line)Describe 2D elements in the canvas. Other 3D objects would not influence their coordinate in the second element.

The first and second element is must, and the third and fourth element is optional.

3.11 Implementation

This section will show the implementation of StoryGenerator.

3.11.1 XML File Generation. The XML file will be created by the SuperSQL through the "*.ssql" file.

Then we will explain Algorithm1. First, input the tree structure data, layout formula through the query in the "*.ssql" file. If the CodeGenerator in the SuperSQL reads the media name of Unity_dv, it will add an XML tag to the output. Then, SuperSQL will read the grouper and generate the grouper node with the attribute of g1(,), g2(!), g3(%), g4(#), g5(&). Next time, read the child node of the grouper, generate the connector node with the attribute of c1(,), c2(!), c3(%), c4(#). The object around the connector will be a child of the connector node.

Algorithm 1 XML File Generation**Require:** tree structure data, layout formula**Ensure:** XML file

```

1: add XML tag
2: for all operator do
3:   if there is the grouper then
4:     create grouper element
5:   end if
6:   if there is the connector then
7:     create connector element
8:   end if
9:   while the function has function in argument do
10:    save information of current function
11:    read next function
12:    if object generation function then
13:      create object element
14:      add saved optional function information as child node
        of object
15:    end if
16:  end while
17: end for
18: return XML file

```

Then read the optional function of the query. If the function is an object generation function, creating the object or asset node. Then read the optional function, add the information to the object or asset node until the last function.

3.12 Object Generation

The Algorithm2 shows how the object is created in Unity. First, the XML reader will read the XML file for all elements. If there is a grouper element, generate the grouper object, a wrapper for the child object, and read its child element. Then if there is a connector element, generate the connector object, a wrapper for the child object, and read its child element. When reading an object generation element, instantiate an object by the information of the object element. Then if the object element has an optional function as a child, add the information to the object until the last child.

4 USE CASE

This chapter shows the use case of storytelling with Unity and SuperSQL.

4.1 Sample Data

The use case uses the weather data about nine dimensions of Tokyo 2020[16] and an asset of farm[14]. Table 2 shows the corresponding relationship between data information and asset property. The environment of the scene is decided through the following description of an environmental function, and Table 3 shows the several environmental functions supported by the system.

Algorithm 2 Object Generation**Require:** XML file**Ensure:** Object with function information

```

1: read XML file
2: for all XML element do
3:   if there is the grouper element then
4:     create grouper object
5:   end if
6:   if there is the connector then
7:     create connector object
8:   end if
9:   while the element has child do
10:    if object generation function then
11:      create object
12:    end if
13:    if optional function then
14:      add optional function information to the object
15:    end if
16:    if environment function then
17:      generate environment manager
18:      find the environment object and assign value
19:    end if
20:  end while
21: end for
22: return Object

```

Table 2: Corresponding relationship of information and asset

Data information	Asset property
air pressure	strength of lighting
max precipitation	size of grass
average precipitation	amount of grass
average temperature	number of animals
min temperature	min size of animals
max temperature	max size of animals
humidity	velocity of the waterfall
wind speed	rotate speed of the windmill
time of sunlight	number of birds (higher air pressure more bird)

Table 3: Environmental Function

Function	Asset property
env('waterfall', 'rotate', 'velocity', value)	velocity of waterfall
env('windmill', 'rotate', 'velocity', value)	velocity of windmill
env('sunlight', 'light', 'intensity', value)	strength of sunlight
env('grass', 'transform', 'scale', value)	size of grass

4.2 Query Example

The following query is executed, and the chapter shows the result of it. -QueryExample

```

GENERATE Unity_dv
scene(farm),
[random('animal', w.tempera_average, w.tempera_min,
  ↳ w.tempera_max),
assert('bird', w.pressure),
optional_annotation(env('waterfall',
  ↳ 'ParticleSystem', 'speed', w.humidity), 'The
  ↳ humidity of' || w.month || 'is' ||
  ↳ w.humidity),
optional_annotation(env('Sun', 'Light',
  ↳ 'intensity', w.sunlight_time), 'The sunlight
  ↳ time of' || w.month || 'is' ||
  ↳ w.sunlight_time),
optional_annotation(env('grass',
  ↳ 'ParticleSystem', 'max_size', w.precip_max),
  ↳ 'The max of precipitation of' || w.month ||
  ↳ 'is' || precip_max),
optional_annotation(env('grass',
  ↳ 'ParticleSystem', 'max_particles',
  ↳ w.precip_total), 'The total of precipitation
  ↳ of' || w.month || 'is' || w.precip_total),
optional_annotation(env('wind', 'rotate',
  ↳ 'velocity', w.wind), 'The strength of Wind
  ↳ of' || w.month || 'is' || w.wind),
w.month
]&

from weather_tokyo w

```

The scheme used in this query shows the following.
-weather(id, pressure, precip_max, precip_total, tempera_average, tempera_min, tempera_max, humidity, wind, sunlight_time);

4.3 Execution Result

This chapter shows the result of execution. Figure 9 shows the dataset of August, and Figure 10 the dataset of December. You can see that animal of August is bigger than in December; also, the sunlight is more bright.

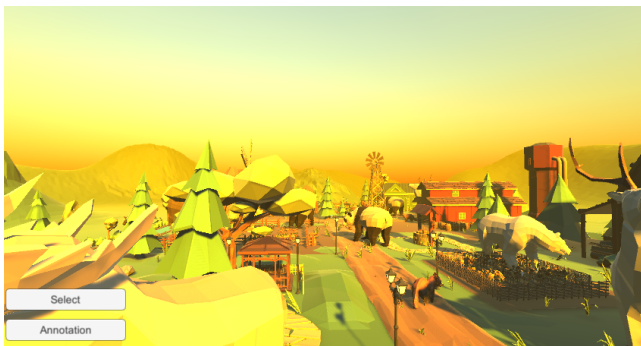


Figure 9: Dataset of August

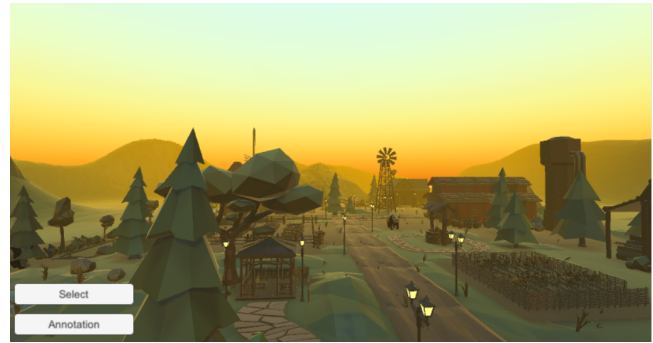


Figure 10: Dataset of December



Figure 11: Example of slideshow repeat

- Annotation
Annotation can be assigned to an object through the optional annotation function shows in Figure 13.
- Slideshow repeat.
Figure 11 shows the example of the slideshow repeat. Grouping with the month, then scenes and corresponding UI buttons of the month will be generated.

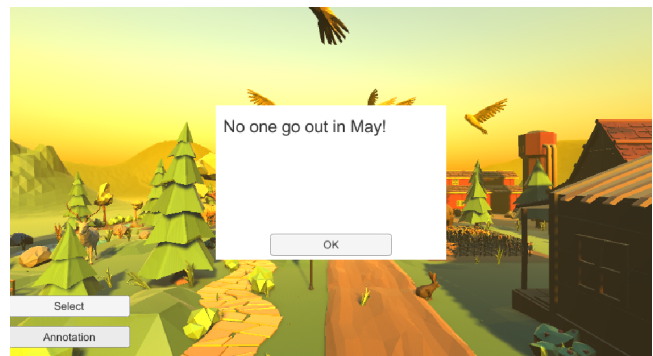


Figure 12: Annotation for scene



Figure 13: Annotation for object

4.4 Another Presentation

SuperSQL can alter the presentation easily by just rewrite the query. For this use case, we change the corresponding relationships of air pressure and sunlight time. Higher air pressure will show a more bright light in a new presentation, and longer sunlight time will bring more birds.



Figure 14: Dataset of August after

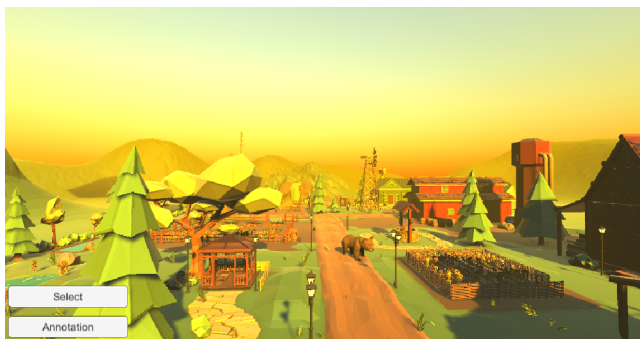


Figure 15: Dataset of December after

Figure 14 and 15 shows the new presentation of the same dataset. It is known that December is more bright than August because

of the higher air pressure. In SSstory, creators can improve their presentation quickly to tell the message expectedly.

4.5 Another Dataset

SSstory can use the same asset to present other datasets. As an alternative, we use the data of C-VOID19 of 2020 in Tokyo, Japan. Table 4 shows the corresponding relationship in a new dataset.

Table 4: Corresponding relationship in Covid-19 dataset

Data information	Asset property
newly infected cases	strength of lighting (less people more bright)
total severe cases	size of grass
newly death cases	amount of grass
total vaccine inoculations	number of birds
newly vaccine inoculations	max size of birds

-QueryExample2

```
Generate Unity_dv
scene('Farm')
, random('Bird', c.total_vaccine/100000, 0.1,
  ↳ c.newly_vaccine/500000 + 0.5)
, env('Directional Light', 'Light', 'intensity',
  ↳ (8000 - c.newly_infection)/3000)
, env('GrassA', 'ParticleSystem', 'startSize',
  ↳ c.newly_death/2)
, env('GrassA', 'ParticleSystem', 'maxParticles',
  ↳ c.total_severe*2)
!annotation('textbottom', c.date || ',' || 'newly
  ↳ infection' || ':' || c.newly_infection)

from covid19 c
where c.date = '2021-3-22'
```



Figure 16: Dataset of 22nd March 2021 about Covid-19



Figure 17: Dataset of 22nd April 2021 about Covid-19



Figure 18: Dataset of 22nd May 2021 about Covid-19

Figure 16 and 17 show the result of visualization. New infected, severe cases, death cases were increased from March to April 2021. Also, the vaccine inoculations were increased through the number of birds. The result of 22nd May (Figure 18) has lush grass also the increasing birds which show the infection condition intuitively. From this visualization work, we can tell a message that even though the virus spread is uncontrolled, the inoculation was increased, and its effect will be expected.

4.6 Another Asset

In this section, we will show an alternative asset for the same dataset of Covid-19. Table 5 shows the corresponding relationship in the Covid-19 dataset with a new asset, an amusement park [15].

Table 5: Corresponding relationship in Covid-19 dataset with a new asset

Data information	Asset property
newly infected cases	strength of lighting (less people more bright)
total severe cases	size of foliage
death cases	amount of foliage
total vaccine inoculations	size of attractions
newly vaccine inoculations	rotate rate of attractions

-QueryExample3

```
Generate Unity_dv
scene('Park')
,env('Bushes', 'transform', 'scale_child',
  ↳ c.total_severe/400)
,env('Trees', 'transform', 'scale_child',
  ↳ c.newly_death/50)
,env('Directional Light', 'Light', 'intensity',
  ↳ (8000 - c.newly_infection)/5000)
,env('FerrisWheel_Rotate',
  ↳ 'FerrisWheel_Rotation', 'speed',
  ↳ c.c.newly_vaccine/1000)
,env('Carousel_Rotate', 'Carousel_Rotation',
  ↳ 'speed', c.c.newly_vaccine/1000)
,env('FerrisWheel', 'transform', 'scale',
  ↳ c.total_vaccine/7000000 + 0.5)
,env('Carousel', 'transform', 'scale',
  ↳ c.total_vaccine/7000000 + 0.5)
,annotation('textbottom', c.date || ',' || 'newly
  ↳ infection' || ':' || c.newly_infection)

from covid19 c
where c.date = '2021-3-22'
```



Figure 19: Dataset of 22nd March 2021 about Covid-19 with a new asset



Figure 20: Dataset of 22nd April 2021 about Covid-19 with a new asset



Figure 21: Dataset of 22nd May 2021 about Covid-19 with a new asset

Figure 19, 20, and 21 show the infection condition of 22nd March, April, and May. We show the range of presentation of this system from these visualization works because users can choose any asset they want to use.

5 EVALUATION

For evaluation, we get ready to invite the experimenter to use this system and compare it with other data storytelling systems in expression, learning cost, and amount of code.

6 CONCLUSIONS

We implement the StoryGenerator to generate a static data video through the timeline grouper at the present stage. We will use the cinematic asset to design and implement the StoryEditor to achieve a high-quality and more accurate data video in the future.

REFERENCES

- [1] Tatsuki Fujimoto, “3D Visualization of data using SuperSQL and Unity”, IDEAS, 2019.
- [2] Unity. <https://unity3d.com/>.
- [3] Edward Segel, Jeffrey Heer, “Narrative Visualization: Telling Stories with Data”, IEEE Transactions on Visualization and Computer Graphics, 2010.
- [4] Fereshteh Amini, Nathalie Henry Riche, Bongshin Lee, “Understanding Data Videos: Looking at Narrative Visualization through the Cinematography Lens”, HAL, 2015.
- [5] Fereshteh Amini, Nathalie Henry Riche, “Authoring Data-Driven Videos with DataClips”, IEEE TRANSACTIONS ON VISUALIZATION AND COMPUTER GRAPHICS, 2017.
- [6] Edward Segel, “Narrative Design Patterns for Data-Driven Storytelling”, IEEE Transactions on Visualization and Computer Graphics, 2010.
- [7] Yanru Lyu, “Visual Data Storytelling: A Case Study of Turning Big Data into Chinese Painting”, HCI 2020.
- [8] Donghao Ren, “ChartAccent: Annotation for Data-Driven Storytelling”, IEEE Pacific Visualization Symposium, 2017.
- [9] Amin Beheshti, Alireza Tabebordbar, Boualem Benatallah, “iStory: Intelligent Storytelling with Social Data”, WWW '20 Companion, 2020.
- [10] Danqing Shi, “Calliope: Automatic Visual Data Story Generation from a Spreadsheet”, IEEE Transactions on Visualization and Computer Graphics, 2020.
- [11] SuperSQL. <http://ssql.db.ics.keio.ac.jp>.
- [12] M. Toyama, “Supersql: An extended SQL for database publishing and presentation”, Proceedings of ACM SIGMOD '98 International Conference on Management of Data, pp. 584–586, 1998.
- [13] BitSplash Interactive, Chart And Graph, <https://assetstore.unity.com/packages/tools/gui/graph-and-chart-78488>.
- [14] Must Have Studio, HappyLifeville.Low Poly Farm, <https://assetstore.unity.com/packages/3d/environments/industrial/happylifeville-low-poly-farm-163983>.
- [15] Must Have Studio, Amuseville. Low Poly Park, <https://assetstore.unity.com/packages/3d/environments/urban/amuseville-low-poly-park-192985>.

- [16] Japan Meteorological Agency. <https://www.data.jma.go.jp/>

Designing a Business View of Enterprise Data

An approach based on a Decentralised Enterprise Knowledge Graph

Max Chevalier

Institut de la Recherche en Informatique de Toulouse
118 route de Narbonne
Toulouse, France 31400
max.chevalier@irit.fr

Franck Ravat

Institut de la Recherche en Informatique de Toulouse
118 route de Narbonne
Toulouse, France 31400
franck.ravat@irit.fr

Joan Marty

umlaut.
3 Bd Henri Ziegler
Blagnac, France 31700
joan.marty@umlaut.com

Bastien Vidé

Institut de la Recherche en Informatique de Toulouse
118 route de Narbonne
Toulouse, France 31400
umlaut
3 Bd Henri Ziegler
Blagnac, France 31700
bastien.vide@irit.fr
bastien.vide@umlaut.com

ABSTRACT

Nowadays, companies manage a large volume of data usually organised in "silos". Each "data silo" contains data related to a specific Business Unit, or a project. This scattering of data does not facilitate decision-making requiring the use and cross-checking of data coming from different silos. So, a challenge remains: the construction of a Business View of all data in a company. In this paper, we introduce the concepts of Enterprise Knowledge Graph (EKG) and Decentralised EKG (DEKG). Our DEKG aims at generating a Business View corresponding to a synthetic view of data sources. We first define and model a DEKG with an original process to generate a Business View before presenting the possible implementation of a DEKG.

CCS CONCEPTS

• **Information systems** → **Enterprise applications**; *Information integration*;

KEYWORDS

Business View, Enterprise Knowledge Graph, Schema Matching

ACM Reference format:

Max Chevalier, Joan Marty, Franck Ravat, and Bastien Vidé. 2021. Designing a Business View of Enterprise Data. In *Proceedings of 25th International Database Engineering & Applications Symposium, Montreal, QC, Canada, July 14–16, 2021 (IDEAS 2021)*, 10 pages.
<https://doi.org/10.1145/3472163.3472276>

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-8991-4/21/07...\$15.00
<https://doi.org/10.1145/3472163.3472276>

1 INTRODUCTION

Today's organisations have large volumes of production data and documents scattered across multiple sources and heterogeneous environments. One of the most popular ways to store this production data is to organise it as "data silos". This type of storage allows companies to organise data according to different criteria specific to their activities (by project, by supplier, by customer, by Business Unit, etc.). Each silo allows local data management with an adapted storage system. However, the isolation of the data contained into silos is a major drawback: this can lead to redundancy or even strong inconsistencies between different silos. Also, the company management staff does not have a Business View of the data due to the isolation of those silos. In addition, a lot of documents generated in the companies are also not intensively used for decision-making.

Different Unified Views construction strategies currently exist, such as Data Warehouses, or Data Lakes. Data Warehouses (DW) are Business Intelligence (BI) databases used to centralise useful data for the decision-makers of an organisation [21]. Data Warehouses contain only a part of production data, which is predetermined and modelled to match specific needs. The integration of this data is named Extract, Transform, Load (ETL). On the other hand, Data Lakes (DL) ingest raw data from multiple sources (structured or unstructured data) and store them in their native format [26]. Also, an advantage of DL is that the data is prepared only when they're used by a user. As opposed to the DW, the Data Lake ingest as much data as possible in the organisation. It also does not require a modelled schema, and can be then used to a greater number of users that are not decision-makers.

The Enterprise Knowledge Graph (EKG) is one of the newest approach that also can be used as a solution to the "data silos problem" in a company [18]. It is defined as a structure used to "represent relationships between the datasets" [8] but also as a "semantic network of concepts, properties, individuals and links representing and referencing foundational and domain knowledge relevant for an enterprise" [11]. An EKG particularly highlights the

relationships existing, for instance, between the currently existing information in the company.

Unfortunately, organisations still have some difficulties to create their own EKG and to integrate in a single schema the large amount of heterogeneous data, information and documents available in the numerous sources they own. Moreover, the obtained model do not really abstract the data to concepts, making its exploitation by decision-makers difficult.

In this paper, we propose an original process to generate a Business View of multiple data sources within an Enterprise Knowledge Graph. The Business View is a synthetic view allowing a non-technical end-user to explore and discover the data of an organisation. Thus, we first detail the different concepts related to our proposal and then describe the generation of such a view through different steps within a Decentralised Enterprise Knowledge Graph.

2 RELATED WORK

Initially, the concept of Knowledge Graph (KG) is defined in the Web Semantic domain. The goal of a KG in Web Semantic is to build an "Entity-centric view" of the data from multiple sources [13]. A KG links multiple resources from different websites together. A resource can be anything, from a webpage to an open data API endpoint that represents an entity. Back in 2012, Google was able to build a Knowledge Graph based on different sources, like Freebase or Wikidata [29].

Outside the Web Semantic domain, a Knowledge Graph may be defined as a graph of entities and relationships [24] or "a network of all kind of things which are relevant to a specific domain or to an organization [...]" [7]. Ehrlinger et al. definition is equivalent to Blumauer's one, except that the possible usage of inference in a Knowledge Graph is added: "A knowledge graph acquires and integrates information into an ontology and applies a reasoner to derive new knowledge" [10]. Referring to these previous definitions, a Knowledge Graph may have technical characteristics but it is specific to a particular domain of research.

An Enterprise Knowledge Graph, a KG applied to an organisation, is a private Knowledge Graph containing private and domain-specific knowledge. The EKG can be used as a Business View that is able to solve the "data silos problem" [18]. The goal of such an EKG is to help the users to represent, manage, exploit the knowledge of an organisation. It also allows machine-to-machine inter-operability using the stored knowledge. The Data Source of the EKG is usually centralized [32].

Some papers explain the implementation of Enterprise Knowledge Graphs [11, 30, 33]. Additionally, some tools allow to integrate database contents [1, 31] or documents [9, 14] into a Knowledge Graph or even directly from texts through Named Entity Resolution [12] and Thematic Scope Resolution [4, 12]. A few less researches have been done around the querying side of Enterprise Knowledge Graphs [30]. Based on the literature, the Enterprise Knowledge Graph (EKG) seems to be a great support for a Business View, as it supports a lot of flexibility. Moreover, relationships may help to understand how the data is related for decision-making processes, and also help how to infer new knowledge from the existing one. However, neither a clear definition of what an Enterprise Knowledge Graph nor a global architecture are defined in the literature,

as far as we know. We also identify that no process to build an EKG schema from the available data exists.

Intending to propose a new way of building Enterprise Knowledge Graphs, we also studied the Schema Matching. The Schema Matching is a set of methods allowing to "determine which concepts of one schema match those of another" [23] in distributed data sources. Schema Matching has been well studied in the literature, both Database Matching [16, 25] aiming at defining methodologies to match different relational databases, and Ontology Matching [3] which aims more at matching web semantic ontologies. New approaches based on similarity have been developed to allow schema matching on graph data structures [22], and improve current Schema Matching techniques [20, 27].

In the next section, we detail our own definition of the Enterprise Knowledge Graph as a Business View of company data. We also define the concept of Decentralised EKG (DEKG). We propose a process to build a DEKG Business View schema from existing data using schema matching, before discussing its architecture and its implementation.

3 ENTERPRISE KNOWLEDGE GRAPH

3.1 From EKG to DEKG

In our context, an Enterprise Knowledge Graph (EKG) represents all the source data of interest for the company in order to offer end users a Business View of this data. Moreover, an EKG should emphasise the relationships that exist between them. Such a Business View allows end-users to identify, locate and access and finally analyse these data. An EKG also helps them in decision-making tasks thanks to the knowledge they can extract from the Business View. Such Business View mostly corresponds to the result of the integration of the different source's schemata. An EKG should be easily **extensible** in terms of data (i.e. should ingest new data sources) and user needs (i.e. queries, exploration capabilities..), without the need of human intervention.

In this paper we define an extension of such a concept that is named a *Decentralised Enterprise Knowledge Graph* (DEKG). The decentralised dimension of a DEKG aims at offering a better scalability of the underlying system since data are not integrated in a centralised system (i.e. only data source schemata are centralised). We based our approach around a global-as-view approach [19].

Furthermore, in contrast with the above definition, an DEKG proposes to end-users a Business View that is a **synthetic view** of source data through a unified schema. It is a schema that corresponds to a **"end-user view"** of the global schema content. It "erases" the specific implementation particularities present in the different data sources. Such a view allows the end-user to understand what information is available, how it's linked and in which database/repository it is stored.

To obtain such a Business View, a DEKG implementation is based on a 3-steps process (see Figure 1):

- the first step aims at generating data sources schemata from the different data sources, one independently to others;
- the second step aims at generating a global schema that is the result of the schema matching of the different data source schemata;

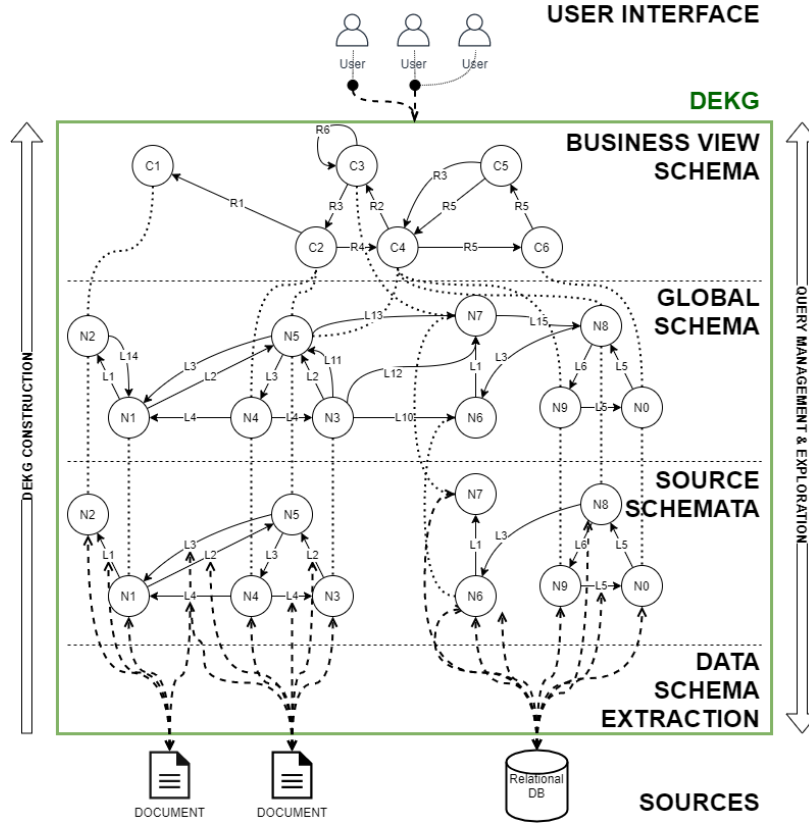


Figure 1: Decentralised Enterprise Knowledge Graph workflow

- the third step is more original since it aims at synthesising the global schema into a synthetic and understandable schema that will be proposed as the Business View to the end-user.

Please note that the DEKG always keeps all the data sources while generating all the different schemata in each step.

Since a DEKG should emphasise the relationships between data and data schema, we propose in this paper to model a DEKG as a set of graphs. These relationships present in the different DEKG schemata, will help decision-making by showing the "context" of the available data. These relationships also support graph exploration or new knowledge discovery.

3.2 Our DEKG construction process

Our process generates different schemata. According to the Figure 1, we define the concepts of source schema, global schema and Business View and explain how to generate them. Moreover, in order to explain every schema generation, we propose an illustrative example.

Example. Three data sources are available (one relational database containing 2 tables and 2 CSV files). The content of each data source is presented in the table 1.

3.2.1 General schema graph model. All the schemata generated by the DEKG is modelled as an heterogeneous graph, based on the Property Graphs [5, 17]. We define a graph g as follows: $g = (V, E)$;

$V = \{\vartheta_1, \dots, \vartheta_v\}$ is the set of nodes and $E = \{e_1, \dots, e_e\}$ the set of edges, with $E \subseteq V \times V$. Each node has a label τ that belongs to $T = \{\tau_1, \dots, \tau_t\}$. The function $w : \vartheta \rightarrow \tau$ is used to return the label τ of a node. Every edge also has a label μ belonging to $M = \{\mu_1, \dots, \mu_u\}$ and the function $n : e \rightarrow \mu$ returns the label μ of an edge. Node and edge labels can be characterised by a set of attributes belonging to $X = \{\chi_1, \dots, \chi_x\}$. Those attributes, in the context of an Enterprise Knowledge Graphs, are also called *properties*. To simplify, we ignore in this definition all companion functions (modifying an attribute, get the list of attribute of an edge or a node...).

Thanks to these definitions, we define every DEKG schemata in the next sections. Due to space limitation, we limit our discussion to structured data sources only.

3.2.2 Source Schema - Step #1. The source schema is composed of all the schemata of all sources handled by the DEKG. This schema results from the extraction of the structure (e.g. list of attributes) of every source independently from others. In order to facilitate schema matching done at step #3 we decided to not store attributes into nodes and choose an "Exploded" Graph. Thus, in the source schema, any node ϑ in V corresponds to either an *entity* (e.g. the name of a table in a relational database), either a *relationship* between two entities (e.g. a foreign key in a relational database), either an *attribute* (property) characterising an entity. The set of node types of V is defined as $T = \{REL, ENTITY, PROP\}$ where "REL"

Table 1: Data Source structures and content

DATABASE									
Students Table				Class Table		Teachers Table			
id	Last Name	First Name	Class (FK)	id	Teacher (FK)	id	Full name		
1	Doe	John	A	A	Alice Machin	1	Alice Machin		
2	Doe	Jane	B	B	Bob Lambda	2	Bob Lambda		
				C	John Doe	3	John Doe		

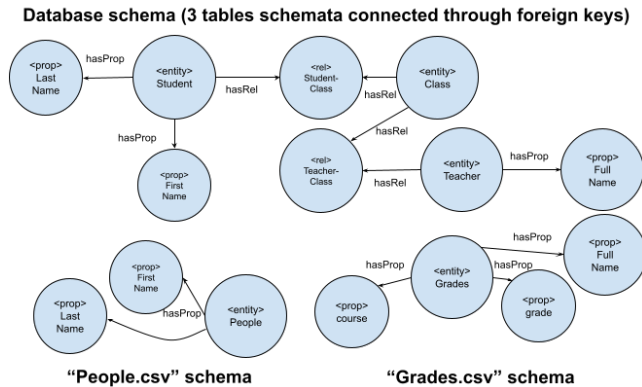
People.csv		Grades.csv		
Last Name	First Name	FULL NAME	GRADE	COURSE
Machin	Alice	John Doe	15	M3xxx
Lambda	Bob	Jane Doe	16	M1xxx
Doe	John			
Doe	Jane			

corresponds to a relationship, "*ENTITY*" to an entity and "*PROP*" to a property.

In order to properly connect the different nodes, we define the different types of any edge e in E as : $M = \{hasProp, hasRel\}$ where "*hasProp*" connects a node of type ENTITY or REL to a node of type PROP and "*hasRel*" connects a node of type ENTITY to another node of type ENTITY.

Moreover, to keep the location of data within the data source, every node or edge are characterised by a minimal set of attributes $X = \{NAME, ID, URIS\}$ where *NAME* corresponds the the name of the entity/property/relationship, *ID* corresponds the identifier of the node/edge and *URIS* corresponds to a set of URIs. Every URI corresponds to the data source URI where the data is located (entity/relationship).

In the following sections and figures, we will represent Exploded Graphs, such as Figure 2, as Properties Graph where our Exploded Graph types $T (< entity >, < rel > \text{ and } < prop >)$ are displayed as labels, and the node attributes are stored as properties (for clarity sake, all the properties, except the name, are not depicted on the figures).

**Figure 2: Data source schemata extraction result**

So, as a result the Source schema contains all the data source schemata that are not yet connected. The objective of the next step

is to construct the global schema. Figure 2 shows the Source schema extracted from our example data sources (Table 1).

3.2.3 Global Schema - Step #2. The global schema corresponds to the result of schema matching of the all data source schemata available in the Source schema.

Inspired from previous work in schema matching, in this paper, we define five additional edge types, meaning that the edge types set M in the global schema are defined as:

$$M = \{hasProp, hasRel, identical, similar, extends, includes, aggregation\}$$

To go deeper in every type, we defined them as following:

- **identical** L_i : can be applied to an edge between two entities, that highlights that they are textually identical and should be treated as exactly the same entity;
- **similar**: can be applied to an edge between two entities, two relationships or two properties, that highlights that they are related and eventually could be treated as the same entity, relationship or property;
- **extends**: can be applied to an edge between two entities, or two relations, that shows that one entity/relation is a "super-class" of another entity/relation. The superclass ontologically represents a more broad entity/relationship;
- **includes**: can be applied to an edge between two properties that highlights that **values** of a property class are included in another property class at a certain rate p ;
- **aggregation**: can be applied to an edge between three or more properties that highlights that a property is a combination of two or more other properties. Somehow, those properties could be treated as similar in a low-granularity view of the schema. Introducing aggregation inside a graph schema makes it an n -uniform hypergraph, as we are linking more than two nodes together.

To infer new edges of such types in the global schema, we also define eight rules that exploit graph structure and data from data sources to create new relationships. These rules and global schema mapping algorithm are detailed in section 3.3.

3.2.4 Business View schema - Step #3. The Business View schema is a schema that abstracts the schema matching process that was

run in the previous steps. At the step, the schema returns to a Property Graph as defined in Section 3.2.1.

The way this Business View Schema is generated is detailed in section 3.4.

3.3 Constructing the DEKG Global Schema

In our process, the main step is the definition of the global schema since it is the result of schema matching of all the source schemata.

In order to connect nodes of the different graphs of source data with new edges we define ten rules that aim at creating a specific type of edge:

$$M = \{hasProp, hasRel, identical, similar, extends, includes, aggregation\}.$$

such rules will allow to construct multiple entities, properties, or relationships and related data. This section details these rules and the way they are applied through a specific algorithm.

3.3.1 Matching schemata rules. To facilitate decision-making on a global view, it's necessary to define new relationships between components of graphs representing source schemata. In our approach, we propose a schema matching based on rules specifying automatically new relationships and/or nodes. We based them on both Database Matching [25] and Ontology Matching [3] ideas to propose both a relational and a semantic approach. Those rules cover multiple approaches of schema matching defined by Rahm et al: Both schema-based and instance-based, and using linguistic and constraints.

These rules are designed to be generic and automatically ran, but they also can be completed by multiple sets of domain-specific rules depending on the information the company wants to include into the Business View.

- **R1:** We create an **"identical"** type edge between two entity nodes if they have a strictly equal value for attribute NAME;

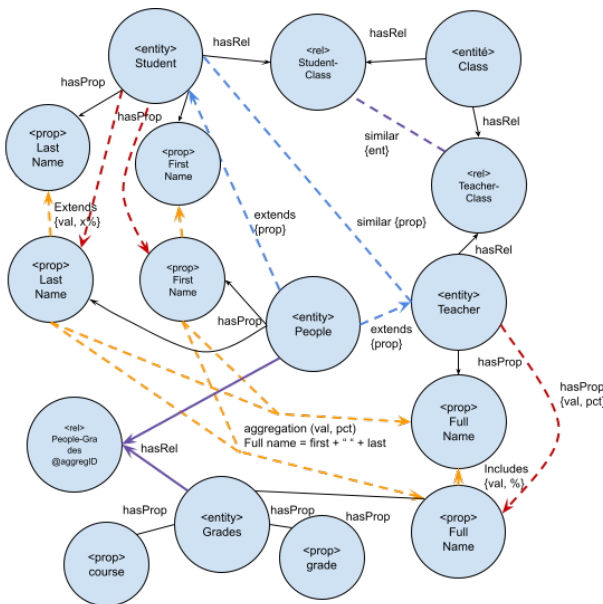


Figure 3: Full matched Global Schema example

- **R2:** We create a "**similar**" type edge between two entity nodes if their NAME are semantically close [15];
- **R3:** We create a "**similar**" type edge between two relationships *A* and *B* if *A* is linked by "*hasRel*" to an entity *C*, *B* linked by "*hasRel*" to another distinct entity *D*, where *C* and *D* are linked by a "similar" edge. *A* and *B* must also be linked to a third common entity *E*;
- **R4:** We create an "**aggregation**" type edge between a set of properties from one entity *A* and one property of another entity *B* if concatenated values of the properties of *A* equal the values of the property of *B*;
- **R5:** We create an "**includes**" type edge from a property *A* to a second one *B* if the values of the property *A* partly equal property *B*. We store the rate of equality in the "include" edge;
- **R6:** We create an "**hasProp**" type edge between an entity *E* linked to a property *A* by "hasProp", and a property *B* of another entity if the property *A* extends *B*;
- **R7:** We create an "**extends**" type edge from one entity (nodes of type ENTITY) *A* to another *B* if all the properties (nodes of type PROP) of *A* extends or aggregates property classes;
- **R8:** We create a new REL node *A*, an "*hasRel*" edge between *A* and an entity *B*, another "*hasRel*" edge between *A* and a different entity *C*, if all properties of *B* are linked by and "*extends*" or "*aggregates*" edge to the properties of *C* and *B* and *C* are not linked by any direct edges (*extends*, *similar*, *identical*, ...).

Algorithm 1: Global Schema Construction: linguistic schema rules

```

1  Input:  $S = s_1, s_2, \dots, s_n$  #Set of all the data source schemata (=
   Source Schema)
2   $GS = \emptyset$ 
3  for each data source schema  $s$  of  $S$  do
4      | copy the data source schema  $s$  into the global schema
   (GS);
5  for each couple of "ENTITY" nodes ( $e1, e2$ ) in  $GS$  do
6      | #R1
7      | if  $e1.name = e2.name$  then
8      | | Create "identical" between  $e1$  and  $e2$  (name) in  $GS$ 
9      | #R2
10     | if  $wordSimilarity(e1.name, e2.name) > simThreshold$ 
   then
11     | | Create "similar" if not exists between  $e1$  and  $e2$ 
   ('name', 'wordsim',  $word-similarity(e1.name,$ 
    $e2.name)$ ) in  $GS$ 
12     | if  $thesaurusSynonym(e1.name, e2.name) > synThreshold$ 
   then
13     | | Create "similar" between  $e1$  and  $e2$  ('name',
   'thesaurus',  $thesaurus.synonym(e1.name, e2.name)$ )
   in  $GS$ 
14  return  $GS$ ;

```


Some of those rules can even create nodes in our Global Schema, meaning that it is possible to create new Entity, Relationship and Property (i.e. a node with one of these type). R8, for instance, is creating a new relationship (node of type REL) between two entities (two nodes of type ENTITY).

The rules are ordered to be executed from R1 to R8 and must be re-executed each time one of the original data source schema changes. The rules have to be executed once the source schema is

Algorithm 2: Global Schema Construction: instance based rules

```

1 Input:  $S = s_1, s_2, \dots, s_n$  #Set of all the data source schemata (= Source Schema)
2 GS = CurrentGlobalSchema #Result of the previous Algorithm
3 for each couple of "REL" ( $r_1, r_2$ ) nodes in GS do
4   #R3
5   # if  $r_1$  and  $r_2$  have a common entity through REL nodes
6   if  $\text{hasRels}(r_1).\text{some}(\text{hasRels}(r_2))$  then
7     if
8        $\text{links}(\text{nodes}(r_1)).\text{filter}(\text{links}(\text{nodes}(r_2))).\text{containsType}(['\text{similar}', '\text{identical}'])$  then
9       Create "similar" between  $r_1$  and  $r_2$  in GS
9 for each set of 2 properties or more ( $p_1, p_2, \dots, p_n$ ) of the same entity and a property  $p_0$  of another entity in GS do
10  #R4
11   $\text{sameNum} := 0$ 
12  for each line of values( $p_1$ ) as  $v_1$ , values( $p_2$ ) as  $v_2, \dots$ , values( $p_n$ ) as  $v_n$  do
13    if  $\text{value}(p_0) \neq v_1.\text{concat}(v_2, v_3, \dots, v_n)$  then
14       $\text{sameNum}++$ 
15  if  $\text{sameNum} > 0$  then
16    Create "aggregation" between  $p_1, p_2, \dots$ , and  $p_n$  with argument ( $\text{sameNum}/\text{length}(\text{values}(p_1, p_2, \dots, p_n))$ ) in GS
17 for each couple of property nodes ( $p_1, p_2$ ) in GS do
18  #R5
19   $\text{includePct} = \text{values}(p_1).\text{some}(\text{values}(p_2))$ 
20  if  $\text{includePct} > 0$  then
21    Create "includes" from  $p_1$  to  $p_2$ , with ( $\text{includePct}/\text{length}(\text{values}(p_1), \text{includePct}/\text{length}(p_2))$ ) in GS
22  #R6
23  if  $\text{links}(p_1).\text{filter}(\text{links}(p_2).\text{filter}(\text{type} = '\text{extends}')).\text{size} > 0$  then
24     $A := \text{nodes}(\text{links}(p_1).\text{filter}(\text{type} = '\text{hasProp}')).\text{filter}(\text{type} = '\text{entity}')[0]$   $B := \text{nodes}(\text{links}(p_2).\text{filter}(\text{type} = '\text{hasProp}')).\text{filter}(\text{type} = '\text{entity}')[0]$  if  $A \neq B$  then
25      Create new "hasProp" between  $A$  and  $p_2$  in GS
26 return GS;

```

Algorithm 3: Global Schema Construction: instance and schema based rules

```

1 Input:  $S = s_1, s_2, \dots, s_n$  #Set of all the data source schemata (= Source Schema)
2 GS = CurrentGlobalSchema #Result of the previous Algorithm
3 for each couple of "ENTITY" nodes ( $e_1, e_2$ ) in GS do
4   #R7
5   if  $\text{links}(\text{nodes}(\text{links}(e_1).\text{filter}(\text{type} = '\text{hasProp}'))).\text{filter}(\text{type} \text{ in } ['\text{includes}', '\text{aggregate}']).\text{includes}(\text{links}(\text{nodes}(\text{links}(e_2).\text{filter}(\text{type} = '\text{hasProp}'))).\text{filter}(\text{type} \text{ in } ['\text{includes}', '\text{aggregate}'])).\text{AND} \text{ links}(\text{nodes}(\text{links}(e_2).\text{filter}(\text{type} = '\text{hasProp}'))).\text{filter}(\text{type} \text{ in } ['\text{includes}', '\text{aggregate}']).\text{includes}(\text{links}(\text{nodes}(\text{links}(e_1).\text{filter}(\text{type} = '\text{hasProp}'))).\text{filter}(\text{type} \text{ in } ['\text{includes}', '\text{aggregate}'])).\text{AND} \text{ "all 2nd arg is 1"} \text{ then}$ 
6     Create "extends" between  $e_1$  and  $e_2$  in GS
7   #R8
8   if  $\text{links}(\text{nodes}(\text{links}(e_1).\text{filter}(\text{type} = '\text{hasProp}'))).\text{filter}(\text{type} \text{ in } ['\text{extends}', '\text{aggregate}']).\text{includes}(\text{links}(\text{nodes}(\text{links}(e_2).\text{filter}(\text{type} = '\text{hasProp}'))).\text{filter}(\text{type} \text{ in } ['\text{extends}', '\text{aggregate}'])).\text{AND} \text{ nodes}(\text{links}(e_1)).\text{some}(e_2) == 0 \text{ then}$ 
9     Create new REL node ( $A$ ) in GS Create new hasRel between  $e_1$  and  $A$  in GS Create new hasRel between  $e_2$  and  $A$  in GS
10 return GS;

```

constructed to start schema matching. An algorithm to construct the global schema is proposed in Algorithms 1, 2 and 3.

Example. After applying the proposed algorithms on Source schema (see Figure 2) we obtain the Global Schema shown on the Figure 3. With only four sources and a few properties per source, we can observe that a lot of new edges (i.e. dashed edges) have been created between nodes from different data source schemata. For instance "red coloured" edges are new hasProp type edges, "purple coloured" edges are new hasRel type edges, whereas as "blue coloured" edges are similar or extends type edges. All those new links will be exploited when constructing the Business View Schema (see section 3.4). The sources URIs of every source node, whatever type they belong to (*entity*, *prop* or *rel*), is stored as an attribute in the matched nodes of the Global Schema. Please note that in Figure 3, we also stored, as an attribute in every element we created during this step, the rule name it has been generated with. That allows the system to differentiate all graph nodes/edges even if they have the same type.

3.4 Constructing the Business View Schema

As explained in section 3.2.4, the Business View Schema is one of the most important aspect of our proposal. That view must be built from the Global Schema, and allow user to see all available data at a glance. The construction of the Business View schema relies on 2 phases.

3.4.1 First Phase. The first phase is quite usual, since it aims to gather all similar nodes/edges available in the global schema into a single node/edge. The result is called "Synthetic Global Schema". It is important to note that the NAME attribute of every gathered nodes is stored in a new attribute (a set of NAME) called *CONTAINS* in the final node. Such new attributes allow us to keep the path to the aggregated nodes within the global schema. Also, every source URIs from the sources nodes are still stored in a new attribute containing the set of URIs corresponding to the sources of those source nodes.

To do so, we exploit the new edges created during schema matching in the Global Schema ; meaning that we gather all nodes connected by edges of type "identical", "similar", "extends", "aggregation" and "includes", as specified in the Algorithm 4.

Algorithm 4: Business View: node combining

```

1 Input: GS
2 for each link R in GS do
3   if type(R.subject) = prop and type(R.object) = prop then
4     if R.predicate = includes(x,y) and x = 1 then
5       Combine R.object into R.subject
6     if R.predicate = aggregate(x,y) and x = 1 then
7       Combine R.object into R.subject
8   if type(R.subject) = entity and type(R.object) = entity
   then
9     if R.predicate = extends then
10      Combine R.object into R.subject Add R.object
      into "contains" attribute in R.subject
11    if R.predicate = similar OR R.predicate = identical
    then
12      Create a node E in GS Set E.name =
      R.subject.name + R.object.name Combine
      R.subject into E Combine R.object into E
13   if type(R.subject) = rel and type(R.object) = rel then
14     if R.predicate = extends then
15      Combine R.object into R.subject Add R.object
      into "contains" attribute in R.subject
16     if R.predicate = identical then
17      Create a node E in GE Set E.name =
      R.subject.name + R.object.name Combine
      R.subject into E Combine R.object into E
18     if R.predicate = similar then then
19       if links(R.subject).filter(type = "hasRel").subject =
      links(R.object).filter(type = "hasRel").subject
      AND links(R.subject).filter(type =
      "hasRel").object = links(R.object).filter(type =
      "hasRel").object then
20       Combine R.object into R.subject Set
      R.subject = links(R.subject).filter(type =
      "hasRel").subject.name + '-' +
      links(R.subject).filter(type =
      "hasRel").object.name

```

3.4.2 Example. When applying such process on the Global Schema (Figure 3) we obtain the resulting Synthetic Global Schema as shown in Figure 4. We can see that the nodes with name "Teacher" and "Student" have been gathered in node with name "People" since *extends* type edges have been created between these nodes in the Global Schema. You can also see that a new attribute named "contains" have been added to node *People* with the values "Teacher" and "Student" to keep a link to the corresponding nodes in the Global Schema.

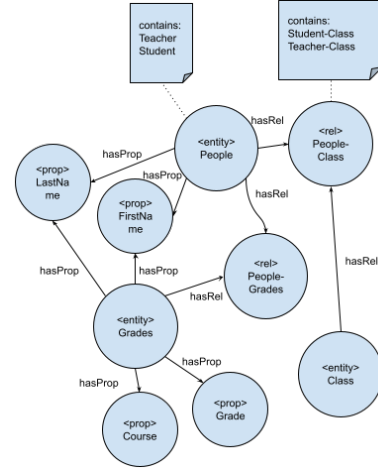


Figure 4: Synthetic global schema

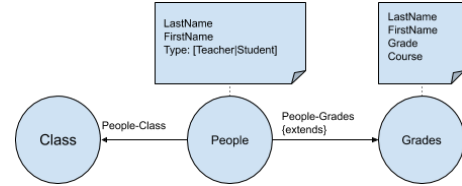


Figure 5: Generated Business View

3.4.3 Second Phase. While the first phase goal was to combine similar/identical nodes of the global schema, the second phase aims at transforming the "Synthetic Global Schema" into the final Business View in which *PROP* nodes are re-integrated in corresponding nodes and relationships as attributes. The Business View is a non-technical view, computed on the fly, that represents all the sources data in a single endpoint. It is aimed at end-users to help them understand and query the available organisations data. The types of the nodes and relationships in the Business View correspond to the NAME attribute value in the Synthetic Global Schema. That phase makes the Business View look like an understandable and regular entity-relationship meta-graph as graph databases like *Neo4j* present their schema. The algorithm 5 describes this nodes re-integration.

Algorithm 5: Business View: schema reducing

```

1 for each "prop" P node do
2   if links(P).length = 1 then
3     N = links(P).nodes[0]
4     for each property Pr of P do
5       N.properties[Pr.name] = Pr
6     Delete links(P)
7     Delete P
8 for each "rel" Rn node do
9   Create a link L
10  L.type = Rn.name
11  for each property P of Rn do
12    L.P = Rn.P
13  Delete links(Rn)
14  Delete Rn

```

3.4.4 Example. After converting the Synthetic Global Schema (Figure 4) into a Business View Schema as explained in above section, we obtain the Business View Schema (Figure 5). As we can see, this figure is quite "simple" and is synthetic enough to be displayed to end-users. Such visualisation will support exploration, navigation or querying operators.

4 IMPLEMENTATION OF A DEKG

Despite Distributed solutions for Knowledge Graphs already exist (like Akutan by Ebay [2]), it has not been clearly defined in academic papers as far as we know, especially for the Enterprise Knowledge Graph. The current well known decentralised one is a federated approach of EKGs allow small subsets of information or knowledge to be linked together, despite being in separated Knowledge Bases [11].

In order to improve maintainability, scalability and high availability of the DEKG we propose a specific architecture (see Figure 6) based on specific DEKG components that can be implemented as services. This proposal is inspired from Federated Databases [28] and its adaptation to the Data Warehouses [6].

The figure 6 follows the workflow of figure 1, from the sources (top of the figure) to the users (bottom). The different components were designed to follow the different steps of the DEKG construction process described at Section 3.2: the Data Component objective to produce the source schema (Step #1; Section 3.2.2) which creates the Sources Schemata; the DEKG Management System is in charge of Step #2 (Section 3.2.3), building and storing the Global Schema; endly the EKG App objective is to build the Business View for the User described in Step #3 (Section 3.2.4), and send queries to the DEKGMS. The following sections introduce those main components of this architecture.

4.1 Data Components.

Our proposition of architecture contains numerous components named "Data Component". Their aim is to interpret the data inside one or multiple data sources which are all related by its location (for instance, in a Business Unit). As we're working in a "Knowledge Graph" environment, the Data Component will have to create a graph schema representing the data of the source. It is also responsible for the link between the schema sent to the "DEKG Management System" and the data contained in the source.

All those Data Components act as "bridges" between the Business View managed by the DEKG Schema component and the data sources. Indeed, when the EKG will be queried, Data Component will be also queried in order to obtain the corresponding data. So, they have to translate the query coming from the Business View to match the queried source (by example, querying a SQL, a document, a CSV...). They all are implemented differently depending on the information the organisation wants to expose for each source. Furthermore, the Data Components ensures the availability to the

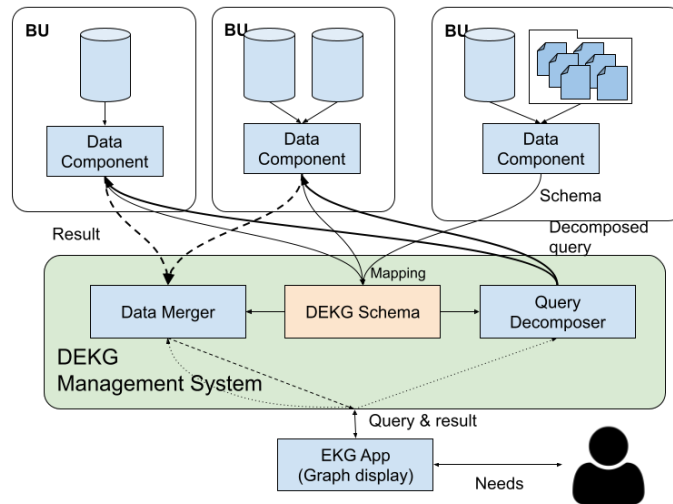


Figure 6: Decentralised EKG architecture components

information. As they're quite small, they can be scaled both horizontally and vertically to ensure both a great scalability and availability to the users. Finally, those components also insure the performance of the overall system. They can for instance cache either data or information, or even queries to answer faster to the queries made by the Decentralised Enterprise Knowledge Graph (DEKG) Management System.

4.2 DEKG Management System.

As shown in Figure 6, a bigger component named "DEKG Management System" is proposed. Its aim is to manage the User and Applications queries and communicate with every Data Component to answer to the user queries. It builds and manages all DEKG schemata and generates at the end the Business View. The answers, the schema, and queries must be transparent to the user as if it was querying a centralised system. The Management System is divided in three essential components : the "Query Decomposer", the "DEKG Schema", and the "Data Merger".

The most important and most complex component of the DEKG Management System is the "DEKG Schema" component. It integrates, stores, and maps the different schemata - not the actual data - coming from all the Data Components included in the Decentralised Enterprise Knowledge Graph. It is in that specific component that the Global Schema of the section 3.3 is constructed and stored.

The **Query Decomposer** is the component receiving the queries from the User/App. Its goal is to break down the user query into sub-queries, which will be sent to the corresponding Data Components using the DEKG schema. The **Data Merger** goal is to get the different responses from the Data Components and merge them back as a single response using the original user query and the DEKG Schema. That unified response is sent to the user.

To do so, the Query Decomposer must use the previous Global Schema to expand the user query into subsets of queries for each source, concurrently run them and all subset responses. Those are received by the Data Merger, which will need to aggregate them, and manage the inconsistencies between all sources [19, 23].

4.3 Querying the DEKG Synthetic View

Finally, the component named "EKG App" in the Figure 6 represents the Human Machine Interface between the end-user and the Decentralised Enterprise Knowledge Graph. This component allows the user to visualise the Business View as specified in the Section 3.4 and shown on Figure 5. The Application goal is also to query the whole DEKG from a unified endpoint.

5 EXPERIMENTAL RESULTS OF THE DEKG

To evaluate our DEKG, we decided to implement real-world open-data on a real use case. Our EKG would allow a user of Toulouse or

its surrounding cities to know which vaccination centre they can go, depending either on their city name, or postal code. In France, a Postal Code can cover multiple cities or a city can have multiple postal codes: thus the importance to be able to search both by "City" or "Postal Code".

To do so, we integrated the french Toulouse city Opendata (Communes Toulouse¹; Code postaux Toulouse²) containing data on all Cities of Toulouse and ZIP Codes; and a COVID-19 dataset from France (Centres de Vaccination³) representing all available vaccination centres in France.

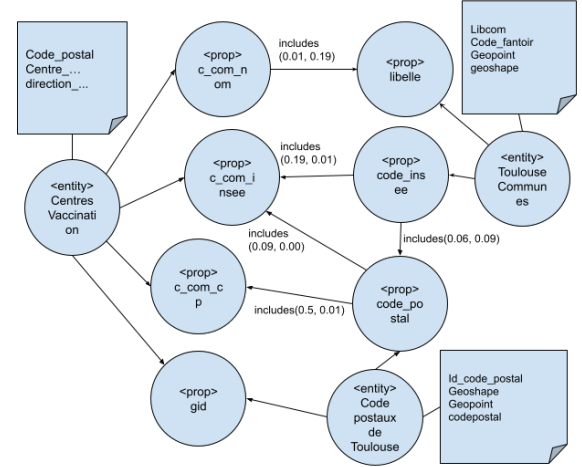


Figure 7: Open Data Schema Matching

As those open-data are all flat files, there were not any relationships in their initial schemata. The only applicable rule in that specific case was the Rule 5 (creating includes between properties). After Global Schema reducing in Figure 7, we can see that we are able to get indirect relationships between entities. That example has shown us that even with if not all rules are applicable on data sources, our approach is able to highlight and create relationships between data sources.

To continue our tests, we ran the algorithm onto our first School example, but also on enterprise employees skills data, acquired with internal survey. All the results of the different steps execution times, and the number of nodes and links contained in both the Global Schema and the Business View are presented on the Table 2. We

¹<https://data.toulouse-metropole.fr/explore/dataset/communes/information/>, accessed 2021-05-11

²<https://data.toulouse-metropole.fr/explore/dataset/codes-postaux-de-toulouse/information/>, accessed 2021-05-11

³<https://www.data.gouv.fr/fr/datasets/reactions-commune-cms/>, accessed 2021-05-11

Table 2: Algorithm runs on different datasets

Dataset	Execution time (ms)					Global Schema		Business View		
	Source Load	Schema Load	Schema Matching	Schema Reducing	Total	Nodes	Links	Nodes	Links	
School Example	6.141	2.75	3.992	1.224	35.221	22	31	5	6	
France OpenData	96.283	29.796	424.383	3.237	563.481	52	56	10	13	
Company Team of Teams	12.362	36.748	1:02.683 (m:ss)	65.805	1:02.807 (m:ss)	281	7766	214	7681	

can see that when the amount of data grows, the Schema Matching is the step taking the most time. Also, the Business View has a much more reduced number of nodes and links compared to the Global Schema; making it much more understandable by the end-user, especially when the data is clean, and therefore successfully matched.

6 CONCLUSION

Organisations really need a Unified Views of their data in order to strengthen their data management. We presented in this paper the Decentralised Enterprise Knowledge Graph as one solution to build a Unified View of the whole organisation data, through our Business View. Thus, we offered a more organisation-oriented definition of the Enterprise Knowledge Graph and a decentralised architecture that can be implemented in an enterprise. Using the sources schemata and schema matching, our Decentralised Knowledge Graph is able to generate a Business View of all the data and data-sources. This approach has been tested against sample data, but also on real-life data from different public sources of multiple providers.

This Decentralised Enterprise Knowledge Graph can be used by organisation to build Enterprise Knowledge Graph which are not copying the original sources, while still allowing its users to query the source data from a single unified point. Our Business View method allow to show the stored Global Schema to non-technical end-users in an easy and understandable way. This DEKG can be used in a lot of applications the Enterprise Knowledge Graph currently used today, for instance building Data Catalogues of organisations.

Despite our architecture being scalable in terms of sources integration, the Global Schema building might grow in computational complexity as the sources grow. Thus, we plan on working on more specific processes to help the Global Schema update when the sources schemata are updated, while keeping the integrity of the schema. Using graph embeddings to enhance our current rule-set is also planned. Also, we'll need to work and integrate existing methods allowing the queries decomposition and responses merging, that we did not describe and evaluate in this current paper. Finally, another future work is to handle data duplication and inconsistency when the same data can be queried from multiple sources. We've worked until now on non-duplicated and consistent data that showed us the feasibility of our DEKG, but we might face difficulties and challenges when working with low-quality data.

REFERENCES

- [1] Neo4j Data Import: Moving Data from RDBMS to Graph.
- [2] Akutan: A Distributed Knowledge Graph Store, May 2019.
- [3] S. Anam, Y. S. Kim, B. H. Kang, and Q. Liu. Adapting a knowledge-based schema matching system for ontology mapping. In *Proceedings of the Australasian Computer Science Week Multiconference*, pages 1–10, Canberra Australia, Feb. 2016. ACM.
- [4] M. Balakrishna, M. Srikanth, and L. Corporation. Automatic Ontology Creation from Text for National Intelligence Priorities Framework (NIPF). *OIC 2008*, page 5, 2008.
- [5] G. Bergami, M. Magnani, and D. Montesi. A Join Operator for Property Graphs. page 9.
- [6] S. Berger and M. Schrefl. From Federated Databases to a Federated Data Warehouse System. In *Proceedings of the 41st Annual Hawaii International Conference on System Sciences (HICSS 2008)*, pages 394–394, Waikoloa, HI, Jan. 2008. IEEE.
- [7] A. Blumauer. From Taxonomies over Ontologies to Knowledge Graphs, July 2014.
- [8] R. Castro Fernandez, Z. Abedjan, F. Koko, G. Yuan, S. Madden, and M. Stonebraker. Aurrum: A Data Discovery System. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, pages 1001–1012, Paris, Apr. 2018. IEEE.
- [9] A. Dal and J. Maria. Simple Method for Ontology Automatic Extraction from Documents. *IJACSA*, 3(12), 2012.
- [10] L. Ehrlinger and W. WöB. Towards a Definition of Knowledge Graphs. In *Joint Proceedings of the Posters and Demos Track of 12th International Conference on Semantic Systems*, page 4, Sept. 2016.
- [11] M. Galkin, S. Auer, H. Kim, and S. Scerri. Integration Strategies for Enterprise Knowledge Graphs. In *2016 IEEE Tenth International Conference on Semantic Computing (ICSC)*, pages 242–245, Laguna Hills, CA, Feb. 2016. IEEE.
- [12] A. Gangemi, V. Presutti, D. Reforgiato Recupero, A. G. Nuzzolese, F. Draicchio, and M. Mongioli. Semantic Web Machine Reading with FRED. *SW*, 8(6):873–893, Aug. 2017.
- [13] J. M. Gomez-Perez, J. Z. Pan, G. Vetere, and H. Wu. Enterprise Knowledge Graph: An Introduction. In J. Z. Pan, G. Vetere, J. M. Gomez-Perez, and H. Wu, editors, *Exploiting Linked Data and Knowledge Graphs in Large Organisations*, pages 1–14. Springer International Publishing, Cham, 2017.
- [14] L. Han, T. Finin, C. Parr, J. Sachs, and A. Joshi. RDF123: From Spreadsheets to RDF. In A. Sheth, S. Staab, M. Dean, M. Paolucci, D. Maynard, T. Finin, and K. Thirunaryan, editors, *The Semantic Web - ISWC 2008*, volume 5318, pages 451–466. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. Series Title: Lecture Notes in Computer Science.
- [15] W. H. Gomaa and A. A. Fahmy. A Survey of Text Similarity Approaches. *IJCA*, 68(13):13–18, Apr. 2013.
- [16] L. Jiang and F. Naumann. Holistic primary key and foreign key detection. *J Intell Inf Syst*, 54(3):439–461, June 2020.
- [17] M. Junghanns, A. Petermann, N. Teichmann, K. Gómez, and E. Rahm. Analyzing Extended Property Graphs with Apache Flink. page 9.
- [18] Kendall Clark. What is a Knowledge Graph, June 2017.
- [19] M. Lenzerini. Data Integration: A Theoretical Perspective. page 15.
- [20] A. Marie and A. Gal. Managing Uncertainty in Schema Matcher Ensembles. page 15.
- [21] Matthias Jarke, Maurizio Lenzerini, Yannis Vassiliou, Panos Vassiliadis. *Fundamentals of Data Warehouses*. Springer Berlin Heidelberg, 2000.
- [22] S. Melnik, H. Garcia-Molina, and E. Rahm. Similarity Flooding: A Versatile Graph Matching Algorithm and its Application to Schema Matching. page 13.
- [23] M. T. Özsu and P. Valduriez. *Principles of Distributed Database Systems, Third Edition*. Springer New York, New York, NY, 2011.
- [24] H. Paulheim. Knowledge graph refinement: A survey of approaches and evaluation methods. *SW*, 8(3):489–508, Dec. 2016.
- [25] E. Rahm and P. A. Bernstein. A survey of approaches to automatic schema matching. *The VLDB Journal*, 10(4):334–350, Dec. 2001.
- [26] F. Ravat and Y. Zhao. Data Lakes: Trends and Perspectives. In S. Hartmann, J. Küng, S. Chakravarthy, G. Anderst-Kotsis, A. M. Tjoa, and I. Khalil, editors, *Database and Expert Systems Applications*, volume 11706, pages 304–313. Springer International Publishing, Cham, 2019.
- [27] T. Sagi. Non-binary evaluation measures for big data integration. page 22.
- [28] A. P. Sheth and J. A. Larson. Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Comput. Surv.*, 22(3):183–236, Sept. 1990.
- [29] A. Singhal. Introducing the Knowledge Graph: things, not strings, May 2012.
- [30] D. Song, F. Schilder, S. Hertz, G. Saltini, C. Smiley, P. Nivarthi, O. Hazai, D. Landau, M. Zaharkin, T. Zielund, H. Molina-Salgado, C. Brew, and D. Bennett. Building and Querying an Enterprise Knowledge Graph. *IEEE Trans. Serv. Comput.*, 12(3):356–369, May 2019.
- [31] Varish Mulwad. *T2LD - An automatic framework for extracting, interpreting and representing tables as Linked Data*. PhD thesis, Faculty of the Graduate School of the University of Maryland, 2010.
- [32] B. Villazon-Terrazas, N. Garcia-Santa, Y. Ren, A. Faraotti, H. Wu, Y. Zhao, G. Vetere, and J. Z. Pan. Knowledge Graph Foundations. In J. Z. Pan, G. Vetere, J. M. Gomez-Perez, and H. Wu, editors, *Exploiting Linked Data and Knowledge Graphs in Large Organisations*, pages 17–55. Springer International Publishing, Cham, 2017.
- [33] B. Villazon-Terrazas, N. Garcia-Santa, Y. Ren, K. Srinivas, M. Rodriguez-Muro, P. Alexopoulos, and J. Z. Pan. Construction of Enterprise Knowledge Graphs (I). In J. Z. Pan, G. Vetere, J. M. Gomez-Perez, and H. Wu, editors, *Exploiting Linked Data and Knowledge Graphs in Large Organisations*, pages 87–116. Springer International Publishing, Cham, 2017.

Analysis-oriented Metadata for Data Lakes

Yan Zhao
IRIT, (CNRS/UMR 5505)
Université Toulouse 1 Capitole
Centre Hospitalier Universitaire
(CHU) de Toulouse
Toulouse, France
yan.zhao@irit.fr

Julien Aligon
IRIT, (CNRS/UMR 5505)
Université Toulouse 1 Capitole
Toulouse, France
julien.aligon@ut-capitole.fr

Gabriel Ferretini
IRIT, (CNRS/UMR 5505)
Université Toulouse 1 Capitole
Toulouse, France
gabriel.ferretini@irit.fr

Imen Megdiche
IRIT, (CNRS/UMR 5505)
Institut National Universitaire
Jean-François Champollion, ISIS
Toulouse, France
imen.megdiche@irit.fr

Franck Ravat
IRIT, (CNRS/UMR 5505)
Université Toulouse 1 Capitole
Toulouse, France
franck.ravat@irit.fr

Chantal Soulé-Dupuy
IRIT, (CNRS/UMR 5505)
Université Toulouse 1 Capitole
Toulouse, France
chantal.soule-dupuy@ut-capitole.fr

ABSTRACT

Data lakes are supposed to enable analysts to perform more efficient and efficacious data analysis by crossing multiple existing data sources, processes and analyses. However, it is impossible to achieve that when a data lake does not have a metadata governance system that progressively capitalizes on all the performed analysis experiments. The objective of this paper is to have an easily accessible, reusable data lake that capitalizes on all user experiences. To meet this need, we propose an analysis-oriented metadata model for data lakes. This model includes the descriptive information of datasets and their attributes, as well as all metadata related to the machine learning analyzes performed on these datasets. To illustrate our metadata solution, we implemented an application of data lake metadata management. This application allows users to find and use existing data, processes and analyses by searching relevant metadata stored in a NoSQL data store within the data lake. To demonstrate how to easily discover metadata with the application, we present two use cases, with real data, including datasets similarity detection and machine learning guidance.

CCS CONCEPTS

• **Information systems** → *Database design and models*;

KEYWORDS

Data Lake, Metadata Model, Analysis-oriented Metadata

ACM Reference Format:

Yan Zhao, Julien Aligon, Gabriel Ferretini, Imen Megdiche, Franck Ravat, and Chantal Soulé-Dupuy. 2021. Analysis-oriented Metadata for Data Lakes. In *25th International Database Engineering & Applications Symposium (IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada)*

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](https://www.acm.org/permissions).

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Association for Computing Machinery.

ACM ISBN 978-1-4503-8991-4/21/07...\$15.00

<https://doi.org/10.1145/3472163.3472273>

2021), July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3472163.3472273>

1 INTRODUCTION

Data Lakes (DL) have emerged as a new solution for Big Data analytics. The ambition of DL is to offer various capacities such as data ingestion, data processing and data analysis. A DL ingests raw data from various sources, stores data in their native format and processes data upon usage [19]. It can also ensure the availability of processed data and provide accesses to different types of end-users, such as data scientists, data analysts and BI professionals.

The major pitfall of data lakes is that they can easily turn into data swamps (i.e. a simple storage space containing all data without any explicit information on them). On the contrary, data analysts or other users need to retrieve and reuse the datasets, associated processes or analyses. To ensure an efficient capitalization of DL, a centralized metadata management system must be deployed. In the literature, several authors propose DL metadata classifications [3, 8, 26], and different metadata systems have been implemented [2, 5, 9]. However, these solutions mainly focus on data management and do not offer metadata on different stages of DL (ingestion, processing and analysis). Regarding data analysis, different models of machine learning (ML) or data mining analysis are proposed by W3C [6] and other authors [12, 18]. These solutions can support data analysts to make better choices on data mining process for one specific dataset, nevertheless, they mainly focus on the phase of data analysis and do not take advantage of data and data processing information across all datasets in data lakes.

Surprisingly, although the ultimate goal of DL is to facilitate data analytics, few solution is proposed to improve the effectiveness and efficiency of the analysts' work through metadata dedicated to support decisional analysis. Therefore, the objective of this paper is to address this lack. For this purpose, information produced during different phases of data analytics has to be taken into account. In particular, these metadata should include: (i) descriptive information about datasets (statistics on attribute values, management of missing values, relationships between datasets and relationship between attributes) and (ii) analytical information about datasets

(performance of implementation and parameters of the executed algorithms).

The contribution of this paper is to propose a metadata model which contains the information produced in the different phases of data analytics in DL and which can facilitate data analysis tasks. The paper is organized as follows. Section 2 details previous works about metadata in an analysis-oriented purpose. In particular, we detail previous works about metadata classifications found in the Knowledge Discovery Database processes (KDD) context [3], the meta-learning context [21] and the data mining context [6, 12, 18]. Section 3 details the metadata model dedicated to data analysis that includes descriptive and analytical information. In Section 4, we detail three algorithms of metadata feeding and we spotlight two use cases of data analysis based on real datasets.

2 RELATED WORKS

Metadata management is essential for data lakes, different authors provide solutions with different emphases. Datasets metadata, including information of each single dataset and the relationships among them, are studied by various authors [2, 4, 13, 24]. Data processing metadata, aiming to improve the efficacy of data preparation, are also studied by different authors [16, 28]. However, to the best of knowledge, data analysis metadata are slightly studied to better manage data lakes.

We have emphasized the importance of metadata management for data lakes in [19] and proposed a basic metadata model in [20], nevertheless, the metadata model is a simple model which mainly focuses on datasets and the analysis-oriented and data processing metadata are not deeply studied. We have completed data processing metadata in [16], and we address in this paper the analysis metadata aspect for our metadata model.

In the literature, the challenge of metadata for data analysis is particularly studied in the field of meta-learning. Meta-learning aims to learn from past experiences [10], in order to improve the effectiveness of ML algorithms. In this paradigm, metadata describe learning tasks (algorithms) and previously learned models (results). Those metadata concern, for instance, hyper-parameters of predictive models, pipelines of compositions or also meta-features. In the literature, two work axes stand out: (a) the first axis is focused on techniques learning from metadata, for example, by transfer learning; in particular, the recent review of [10] summarizes the different techniques that can be investigated, (b) the second axis is focused on using metadata in different application contexts. For example, we find approaches better supporting the model selection [14] or the recommendation of predictive model hyperparameters [15, 22]. Other recent scenarios are emerging in the context of XAI (eXplainable Artificial Intelligence) such as the approach of [27] which proposes a method to explain how meta-features influence a model performance and expose the most informative hyperparameters.

In a high-level view of this work, a lack of consensus emerges on which metadata to use, not only in the context of ML, but also in data analysis more generally. The works that defines the classification of those metadata in a formal setting are scarce. To the best of our knowledge, the works of [3, 12, 18, 23] and a W3C ML model [6] investigate a classification of metadata related to data analysis.

The authors of [3] give a general overview of different types of necessary metadata to support a KDD process. They globally identify the roles and types of the KDD metadata. In particular, they propose a metadata classification taxonomy covering different types of metadata: general measures, statistical, landmarking measures. However, the classification of metadata is defined in a high level, it needs to be detailed. Moreover, dataset is one of the main element for data analysis, no specific metadata is given to better characterize it. On the other hand, the authors of [23] propose a classification of metadata including simple, statistical, information-theoretic, model-based and landmarking information. In addition to their classification, they propose an R package named MFE¹. This paper has the disadvantage of focusing only on dataset features and it limits its contribution to a classification of metadata without modeling. Paper [3] is centered at the conceptual level whereas the paper of [23] is situated at the technical level. Regarding these limitations, our objective is to cover both conceptual and implementation levels.

Regarding ML model, different ontologies of data mining are proposed [12, 18]. [12] contains detailed descriptions of data mining tasks, data, algorithms, hyperparameters and workflows. [18] defines data mining entities in three layers: specification, implementation and application. The W3C ML model [6] is oriented to ML process. Nevertheless, the relationships between datasets and attributes are not considered. However, one of the important advantages of data lakes is to cross multiple sources of data, so that the relationships between these data are crucial. Hence, adding relationship metadata is inevitable in order to facilitate the work of analysts by selecting and querying relevant datasets and corresponding attributes. Our model contains not only data modeling and algorithm metadata, but also the metadata generated during the data collection and preparation phases, including characteristics, statistical metadata, and relationships between datasets and between attributes. This information is essential to deal with the Feature Selection (FS) [25] problem. It also facilitates data analysis tasks by allowing users to broaden their perspective on the data by viewing the results of performed analyses [1]. A detailed description of our model is discussed in the next section.

3 A GENERIC MODEL OF ANALYSIS-ORIENTED METADATA

The process of data analytics can be described by different phases: business requirement, data exploration, data preparation, modeling/algorithms and data product [11, 17]. In order to simplify data analytics, we propose to manage the metadata which are applied mainly on the following phases : (i) **Data exploration**, which is about collecting and exploring the data that users need. (ii) **Data preparation**, which is about transforming, filtering and cleaning data according to users needs. (iii) **Modeling/algorithms**, which consists in choosing a modeling technique, building a model and evaluating the built model.

To facilitate the data exploration phase, metadata should help users to find datasets that match the analysis objectives or suggest relevant datasets to enrich the analysis. To facilitate the data

¹<https://CRAN.R-project.org/package=mfe>

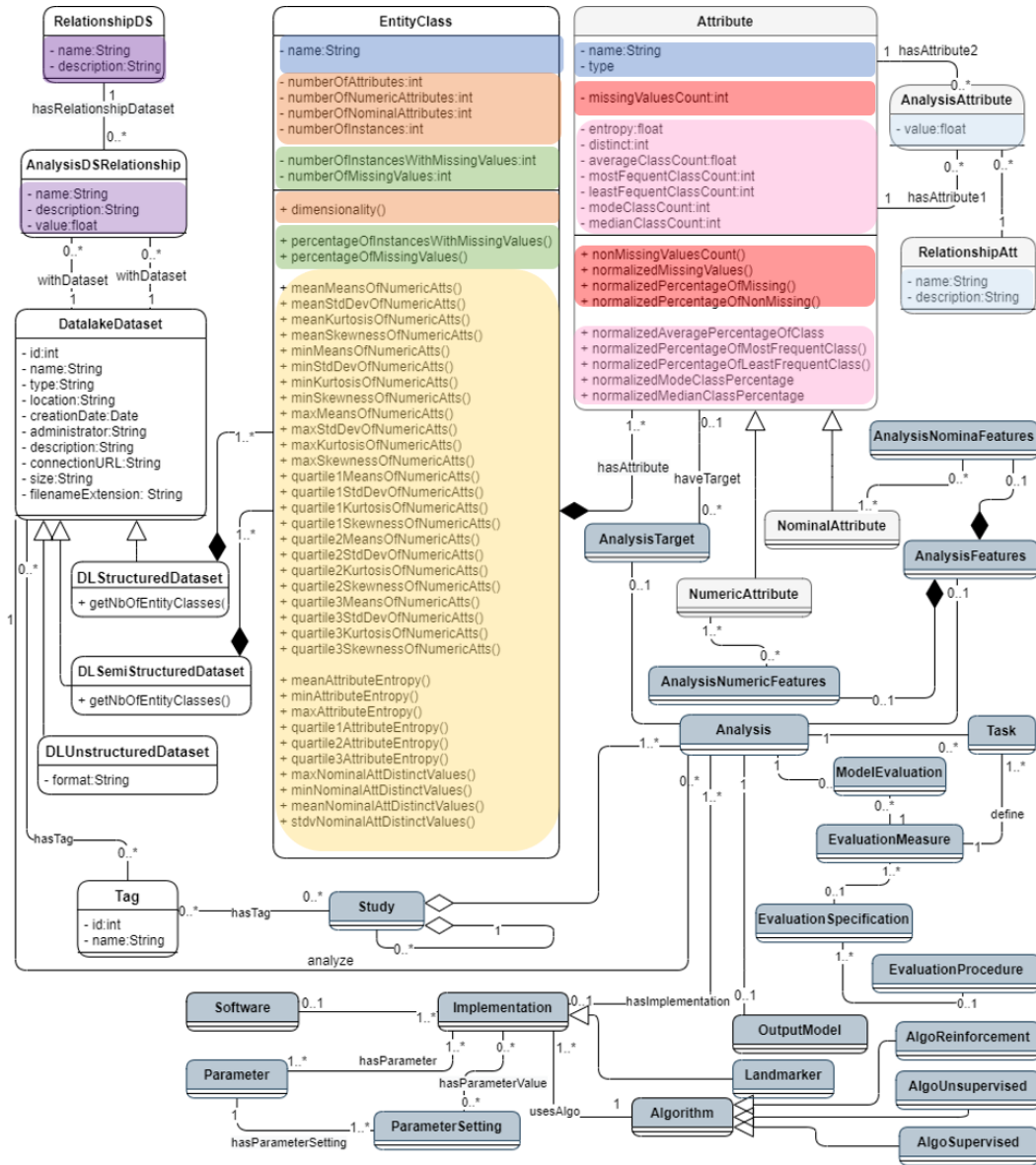


Figure 1: Metadata on datasets and attributes

preparation phase, metadata should present the data characteristics to help users to understand the nature of data and to choose the features of data modeling. Moreover, process characteristics, process definition, technical and content information of data processes should be recorded to help other users understand how data are prepared. To facilitate data modeling phase, metadata should present to users all the existing analyses and certain landmarker results that are carried out on the selected dataset to help users to choose the most appropriate algorithms.

To answer the above mentioned requirements, we extend our previous basic metadata model [20] with the metadata that capitalize the full experiences of data analytic. The complementary metadata can be classified into three categories: datasets, attributes

and analytical metadata. In what follows, we give more details about the different types of metadata in our model. Note that in this paper, we only introduce the analysis-oriented metadata, a more detailed metadata model is available on GitHub².

3.1 Metadata on Datasets

Metadata on datasets allow users to have a general vision of a dataset and can facilitate all data exploration, preparation and modeling phases. They meet the requirements of data analysts who need to know, among other things, (i) dataset structure / schema, which

²https://github.com/yanzhao-irit/data-lake-metadata-management-system/blob/main/images/metadata_model_complet.png

attribute entropy, kurtosis of numeric attributes and standard deviation of numeric attributes. (v) **Relationships** (marked in purple) with other datasets refer to relationships that can be established by users, such as similarity/dissimilarity and correlation, users can define their own algorithms for different relationships calculating.

3.2 Metadata on Attributes

Metadata on attributes allow users to have detailed information of attributes for the chosen dataset and these metadata can also facilitate all data exploration, preparation and modeling. We propose the attribute type, completeness, distribution of values and attributes relationship metadata to answer to the same user requirements at the attribute level. For instance, when users analyze a dataset, they may need to choose particular types of attributes for feature engineering, they may need to choose attributes containing no missing values for predictive models, or when dimensionality reduction approaches (Principal Component Analysis (PCA)) are employed, users need to know the correlation or covariance indicators.

Therefore, for each attribute, we provide the metadata (see the gray classes in Fig. 1) of (i) its **type** (classes *NominalAttribute* and *NumericAttribute*), (ii) the **missing values** (marked in red) which consist of the number of missing values, the number of non-missing values, the normalized missing values count and their percentages, (iii) the **distribution of values** (marked in pink), such as the entropy, distinct values, Kurtosis of the numerical attribute and whether the values of a nominal attribute follow a discrete uniform distribution, and (iv) the **relationships** between two different attributes (marked in light blue), for which we can devise our own relationships such as the Spearman's rank correlation, Pearson's correlation and mutual information.

3.3 Metadata on Data Analysis

As far as machine learning is concerned, especially supervised ML, metadata on datasets and attributes are useful but not sufficient. Indeed, the same dataset can be used for different analyses with different classes or features. Moreover, a user can also be interested in landmarker information which is obtained from performing basic learning algorithms. In addition, it is important to help users to obtain information on previous analyses, as existing analyses can be directly reused or contribute to improve current analyses. Thus, analytical metadata have to ensure collaborative capabilities.

Therefore, to complete our metadata model, we add analytical metadata (see the dark blue classes in Fig. 2) to help users find and possibly reuse existing analyses with their business objectives, selected features and target class, implemented parameters, output model and evaluation of the result : (i) the **study** metadata (marked in red) which is a project containing different analysis, (ii) the **feature analysis** (marked in purple) which concerns the number of features type, the relationships with target attribute and the distribution of all the feature values, (iii) the **target attribute analysis** (marked in blue) in the case of supervised ML, (iv) the **implementation** (marked in green) which is carried out for the analyses, and (v) the **model** of an analysis (marked in orange). (vi) the **evaluation of analysis** (marked in yellow) with different measures. Note that in this paper, we define the class attribute of

supervised analyses as the target attribute, all other attributes are features.

4 IMPLEMENTATION AND USE CASE VALIDATION

The proposed metadata can be used in decision support for data exploration, data preparation (including feature engineering) or the use of model/algorithms. The metadata facilitates (i) the choice of the most relevant datasets and their understanding as well as the feature engineering and (ii) the verification of previous analyses to better prepare and optimize the work.

We chose to use a graph database (Neo4j) to store these metadata for the reason that: (i) a good flexibility can be ensured, (ii) it is easier to query a graph database when the search depth is important, and (iii) some machine learning algorithms are integrated which can be useful to build a recommender system³. Note that the proposed metadata model can also be stored with other technologies, for instance, if relational database is chosen, for the following three algorithms, all the *properties* become *attributes*, all the *nodes* become *instances of the corresponding class table*, and all the *relationships* become *instances of the relationship table*.

This metadata database is integrated in the data lake system and is managed through an application⁴. In the following subsections, we explain how to feed metadata via an application into a graph database by three algorithms. Then we expose two use cases that demonstrate the contribution of our model compared to other models in the literature.

4.1 Metadata feeding

The metadata that we introduced previously are managed by a locally deployed application that interfaces with user input and calls three algorithms to instantiate the Neo4j database according to our metadata model.

In this application, users can input descriptive information, for instance, name and description of datasets or processes, chosen algorithms and set parameters, with an interface. The application can generate automatically other metadata, for instance, schematic metadata of datasets and relationships between different datasets to facilitate future analyses.

During data preparation phase, when a new dataset is stored in the DL (by data ingestion or processing), the application generates descriptive metadata of dataset and its attributes automatically (see Algo. 1).

During data analysis phase, for each study, users can firstly choose features and the target attribute to let the system calculate statistical metadata of features and run landmarkers for machine learning guidance (see Algo. 2). With the information that the system pre-analyzed, the user can implement different algorithms with different parameter settings. For each algorithm with a parameter setting, an *Analysis* and an *Implementation* are instantiated (see Algo. 3).

³<https://neo4j.com/product/graph-data-science-library/>

⁴<https://github.com/yanzhao-irit/data-lake-metadata-management-system>

Algorithm 1: DatasetAttributesMetadata

```

Input: connectionURL, descriptionDS
Data: newDS
/* use a table to store properties */
1 datalakeDatasetProperties ← createProperties('description',
  getDatasetName(newDS), 'connectionURL', 'connectionURL', 'size',
  getFileSize(newDS))
2 datalakeDataset ← createNode('DatalakeDataset',
  datalakeDatasetProperties)
/* store schematic metadata */
3 datasetStruct ← getDatasetStructurality(connectionURL)
4 if (datasetStruct.type = "structured" or "semi-structured") then
5   entities[] ← getEntityClasses(newDataset)
6   foreach e ∈ entities do
7     atts[] ← getAttributes(e)
8     entityClassProperties ← createProperties('name', getEntityName(e),
      getEntityStatistics(atts[]))
      // function getEntityStatistics() returns an array including the name
      and value of each statistical metadata
9     entityClass ← createNode('EntityClass', entityClassProperties)
10    createRelation('DatalakeDataset-EntityClass',
      datalakeDataset, entityClass)
11    foreach att ∈ atts[] do
12      if getAttType(att) = 'numeric' then
13        numericAttributeProperties ← createProperties('name',
          getAttName(att), getNumericAttStat(att)) attribute ←
          createNodeNeo4j('NumericAttribute',
            numericAttributeProperties)
14      else
15        nominalAttributeProperties ← createProperties('name',
          getAttName(att), getNominalAttStat(att))
          attribute ← createNode('NominalAttribute',
            nominalAttributeProperties)
16      createRelation('EntityClass-Attribute', entityClass, attribute)
17      /* For each predefined RelationshipAtt we calculate the value of
        relationship between attributes */
18      analysisAttributes[] ← getAnalysisAttribute(atts[],
        relationshipAtts[])
19      foreach an ∈ analysisAttributes[] do
20        relationArr ← createNode('AnalysisAttribute',
          createProperties('value', an.value))
          createRelation('AnalysisAttribute-Attribute',
            relationArr, an.attribute1)
21        createRelation('AnalysisAttribute-Attribute',
            relationArr, an.attribute2)
22        createRelation('AnalysisAttribute-RelationshipAtt',
            relationArr, an.relationshipAtt)
23      else
24        addAttToNeo4jNode(datalakeDataset, getDatasetFormat(newDS))
25        /* For each predefined RelationshipDS we calculate the value of relationship
        between datasets */
26        analysisDSRelationships[] ←
          getAnalysisDSRelation(datalakeDataset, datalakeDatasets[], relationshipDSs[])
27        foreach anDs ∈ analysisDSRelationships[] do
28          relationDs ← createNode('AnalysisDSRelationship',
            createProperties('value', anDs.value))
          createRelation('AnalysisDSRelationship-DatalakeDataset',
            relationDs, anDs.datalakeDataset1)
          createRelation('AnalysisDSRelationship-DatalakeDataset',
            relationDs, anDs.datalakeDataset2)
          createRelation('AnalysisDSRelationship-RelationshipDS',
            relationDs, anDs.relationshipDS)
    
```

Algorithm 2: Pre_analysis

```

Input: studyName, studyDesc, datalakeDataset, targetAtt, features[],
  analysisName, analysisDesc
Data: selected dataset ds
/* calculate statistics for analysis */
1 Function statisticsAnalysis(study, datalakeDataset, targetAtt,
  features[], analysisName, analysisDesc):
2   analysis ← createAnalysis('Analysis', createProperties('name',
    analysisName, 'description', analysisDesc))
3   createRelation('Analysis-DatalakeDataset', analysis, datalakeDataset)
4   createRelation('Analysis-Study', analysis, study)
5   analysisTarget ← createNode('AnalysisTarget')
6   createRelation('Analysis-AnalysisTarget', analysis, analysisTarget)
7   createRelation('AnalysisTarget-Attribute', analysisTarget, targetAtt)
8   analysisFeatures ← createNode('AnalysisFeatures',
    createProperties(getStatAnalysisFeatures(features[])))
9   createRelation('Analysis-AnalysisFeatures', analysis, analysisFeatures)
10  analysisNumericFeatures ← createNode('AnalysisNumericFeatures',
    createProperties(getStatAnalysisNumericFeatures(features[])))
11  createRelation('AnalysisNumericFeatures-AnalysisFeatures',
    analysisNumericFeatures, analysisFeatures)
12  analysisNominalFeatures ←
    createNodeNeo4j('AnalysisNominalFeatures',
    createProperties(getStatAnalysisNominalFeatures(features[])))
13  createRelation('AnalysisNominalFeatures-AnalysisFeatures',
    analysisNominalFeatures, analysisFeatures)
14  return analysis
/* run predefined landmarks */
15 Function
  runLandmarkers(study, datalakeDataset, targetAtt, features[]):
16  landmarks[] ← getLandmarkers()
17  foreach lm ∈ landmarks[] do
18    analysis ←
      statisticsAnalysis(study, datalakeDataset, targetAtt, features[], lm.name,
        lm.description)
19    output ← runAlgo(analysis, lm)
20    outputModel ← createNode('OutputModel', createProperties('name',
      'description', output))
21    createRelation('Analysis-OutputModel', analysis, outputModel)
22    foreach m ∈ getEvaluationMeasures() do
23      modelEvaluation ← createNode('MoedlEvaluation',
        createProperties('value', evaluate(m, analysis)))
24      createRelation('Analysis-ModelEvaluation',
        analysis, modelEvaluation)
25      createRelation('ModelEvaluation-EvaluationMeasure',
        modelEvaluation, m)
26
27  study ← createNode('Study', createProperties('name', studyName,
    'description', studyDesc))
28  runLandmarkers(study, datalakeDataset, targetAtt, features[])
    
```

Algorithm 3: Analyse_dataset

Input: *study*, *analysisName*, *analysisDesc*, *datalakeDataset*, *targetAtt*, *features*[], *sourceCode*, *algo*, *paras*[], *paraSets*[]

Data: selected dataset *ds*

```

1 analysis ← statisticsAnalysis(study, datalakeDataset, targetAtt,
  features[], analysisName, analysisDesc)
2 implementation ← createNode('Implementation',
  createProperties('sourceCode', sourceCode))
3 createRelation('Implementation-Analysis', implementation, analysis)
4 algo ← createNode('Algorithm', createProperties('name', algo.name,
  'description', algo.description))
5 createRelation('Implementation-Algorithm', implementation, algo)
6 foreach para ∈ paras[] do
7   parameter ← createNode('Parameter', createProperties('name', para))
8   createRelation('Implementation-Parameter', implementation, parameter)
9 foreach paraSet ∈ paraSets[] do
10  parameterSetting ← createNode('ParameterSetting',
  createProperties('value', paraSet.value))
11  createRelation('Implementation-ParameterSetting',
  implementation, parameterSetting)
12  createRelation('ParameterSetting-Parameter', parameterSetting,
  getParameter(paraSet, para))
13 output ← runAlgo(analysis, algo)
14 outputModel ← createNode('OutputModel', createProperties('name', ,
  'description', output))
15 createRelation('Analysis-OutputModel', analysis, outputModel)
16 foreach m ∈ getEvaluationMeasures() do
17  modelEvaluation ← createNode('ModelEvaluation',
  createProperties('value', evaluate(algo, analysis)))
18  createRelation('Analysis-ModelEvaluation', analysis, modelEvaluation)
19  createRelation('ModelEvaluation-EvaluationMeasure',
  modelEvaluation, m)

```

4.2 Use Case Validation

In this section, we illustrate two examples based on real-life situations to show the utility of stored metadata.

As a running example, a data analyst in biology, named Bob, would like to identify the indicators having an important impact on the colon cancer. For this purpose, he uses the *CHSI*⁵ dataset including more than 200 health indicators for each of the 3,141 United States counties. A data engineer constructs three extracts of the dataset to prepare an analysis of three cancers (colon, breast and lung) from this initial dataset according to the objective of binary classification (to have or not the cancer type). Indeed, due to the high number of indicators dedicated to multiple types of diseases (especially on cancers), an analysis can be difficult to be performed directly in a single dataset.

These datasets have six indicators oriented to the analysis of the *colon*, *breast* and *lung cancers*, respectively, of people with the measures of *obesity*, *high blood pressure* and *smoker*. There are 3,141 rows in the dataset and each of them represents a group of people living in a same *county* for each *state* of the United States.

4.2.1 Use Case 1: Dataset similarity detection. Identifying the right ML model for a given dataset can be a very tedious task. On this way, detecting similar datasets where previous models have

already been executed is a possible solution. This would make it easier for a user to identify the type of algorithm that can be launched for their dataset. This approach is inspired by the automated ML field where a model is automatically proposed from a set of existing datasets and ML workflows [7]. Thanks to the dataset and attribute metadata, it is easier to detect stored close datasets, as long as a proximity measure has been implemented in the ML (*Relationship* metadata in Fig. 1).

In our running example, a dissimilarity measure, proposed in [21], is applied (Note that other different algorithms can be also generated). In different analysis contexts, users can use different algorithms. This measure is defined on two levels. The first level estimates a dissimilarity between datasets (classes *AnalysisDSRelationship* and *RelationshipDS*) and the second level is between attribute (classes *AnalysisAttribute* and *RelationshipAtt*).

Bob wants to retrieve more datasets which are close to the colon cancer dataset that he analyzes. Thus, he uses the metadata management application to find colon cancer dataset, and goes to the *Relationship Dataset* tab to check the information that he needs (see left image in Fig. 3). He finds out that breast cancer dataset and lung cancer dataset are both similar to the colon cancer dataset (with the dissimilarities <0.01).

Thanks to the result, Bob decides to enrich his analysis with the breast and lung cancer datasets. Moreover, Bob can searching all the analyses/algorithms that are already executed on these datasets. For each algorithm, he can get the information of used algorithm name, parameter names, parameter values and evaluation values (see right image in Fig. 3). For instance, the shown result concerns an analysis of breast cancer, for which a random forest algorithm is preformed with two parameters (*n_estimators* = 10, *test_size* = 0.2) and the result is evaluated by accuracy (0.42).

Looking at these results, Bob decided to perform a Random Forest and SVM algorithms on his dataset because they give the best results in terms of accuracy on the breast (0.48) and lung cancer (0.58) analyses.

4.2.2 Use Case 2: Machine learning guidance. Metadata based on landmarks are a powerful solution to estimate which kind of machine learning model will be performed better on a dataset. By running a set of diverse simple models on the dataset, a baseline of model performances can be established. This baseline is a first indicator of possible performances for more complex versions of each model, which reduces the need for trial and error testing commonly associated with the choice of machine learning model.

In our running example, four landmarks (and derived versions using different parameters) are available in the Data Lake. These landmarks include: Decision Tree, Naive Bayes, KNN and Random Forest (see class *Landmarker* in Fig. 2). They are evaluated thanks to one measure: Error rate.

Bob wishes to execute a predictive model on the colon cancer dataset. To guide him, he wants to know the error rate of the landmarks available in the data lake. In the application, he can check the existing landmarks and evaluations one by one. For instance, the left image of Fig. 4 concerns the landmark *RandomTreeDepth3* and its error rate is 0.4649; the right image of Fig. 4 concerns the landmark *REPTreeDepth3* and its error rate is 0.5070.

⁵<https://healthdata.gov/dataset/community-health-status-indicators-chsi-combat-obesity-heart-disease-and-cancer>

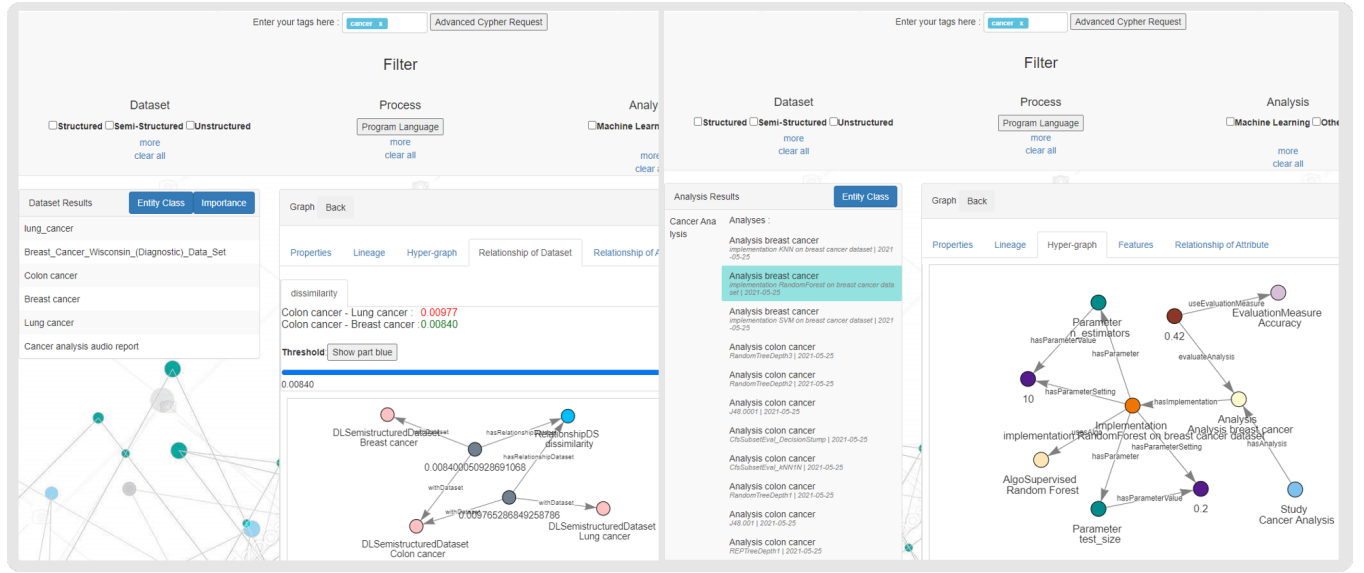


Figure 3: Application results for the use case 1

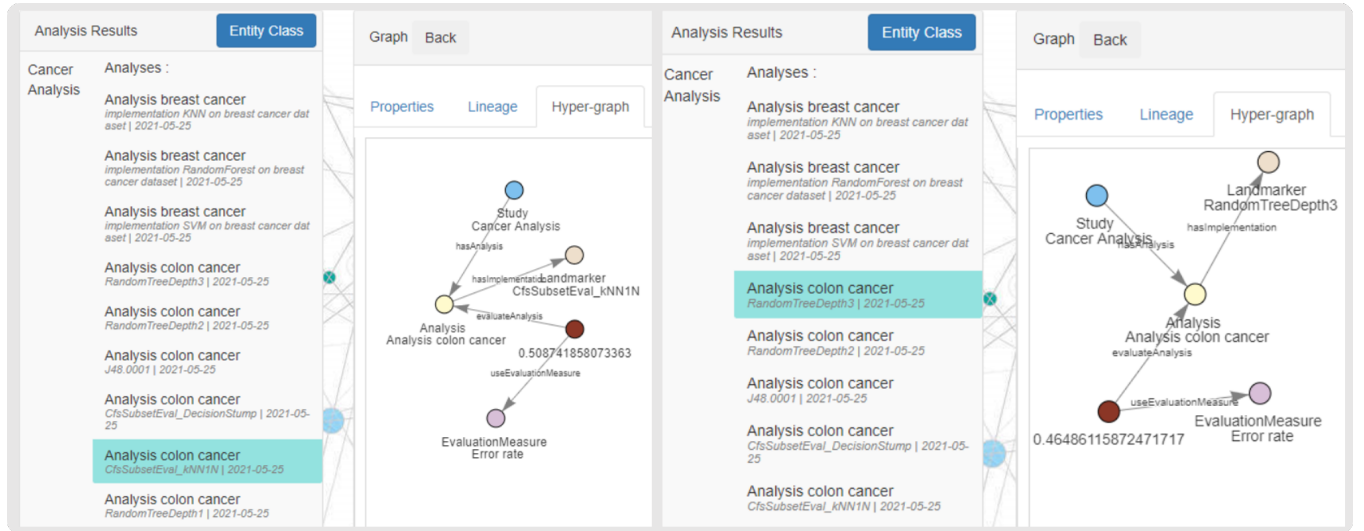


Figure 4: results for the usecase2

After checking all the landmarks that have been executed on the colon cancer dataset, Bob finds the lowest error is performed by *J48.0001*. Therefore, Bob can use the landmark *J48* which gives him a basis to build a better model then.

The application has two accesses, one is a graphical interface that dedicated to all users of the data lake (data scientists, statisticians and analysts) for helping them find, access and reuse existing datasets or analyses. For specialists who have the skills of Neo4j, we provide a second access for more rich research with more characteristics. If Bob does not want to click on every landmark to look for the lowest error rate, he can always do advanced research

by writing query by himself. We provide a function of free consultation in the wapplication which requires Cypher (Neo4j querying language) skill. So that Bob can use the Chyper query below to find all the landmarks of colon cancer dataset (left result in Fig. 5).

```
MATCH
  (ds) -[rhe : hasEntityClass] -> (ec :
    EntityClass) ,
  (a : Analysis) -[ra : analyze] -> (ds) ,
  (a) -[rhi : hasImplementation] -> (lm :
    Landmarker) ,
```

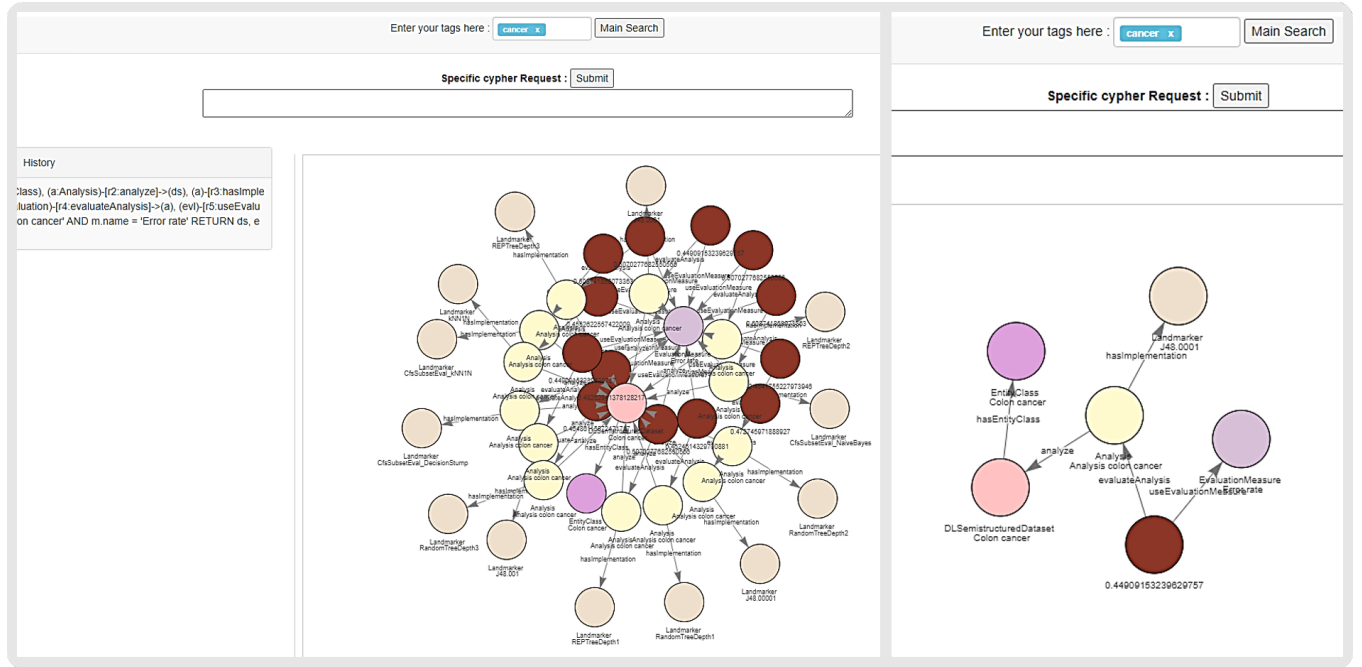


Figure 5: Query and result of the free search

```
( evl : ModelEvaluation ) -[ re :
    evaluateAnalysis ] -> ( a ) ,
( evl ) -[ ru : useEvaluationMeasure ] -> ( m )
WHERE
    ds . name = ' Colon cancer '
    AND m . name = ' Error rate '
RETURN ds , ec , a , lm , evl , m , rhe , ra ,
    rhi , re , ru
```

By adding one condition on the query, Bob can get the lowest error rate landmarker directly (right result in Fig. 5).

```
ORDER BY evl . value ASC LIMIT 1
```

4.3 Use case discussions

Machine learning guidance and finding similar datasets are actual topics in the literature but tackled separately. In this section we introduced the solution to answer these needs through the use cases. Our model covers complete panel of metadata that could be used for different use cases. If we recall the interest of metadata to prevent the data lake from reverting to data swamp, there is inevitably a discussion to be conducted on the time saved by having such a metadata system implemented. Thanks to our metadata model, Bob could automatically access to various datasets and their statistics information as well as the already performed algorithms. During the use case 1, Bob queried pre-calculated correlations during a feature selection task; In the use case 2, the data lake brings up the most relevant algorithms in front of a new dataset. Without the data lake metadata, Bob would have had to manipulate the correlation functions himself or use specific tool or programs for obtaining

landmarkers results and handling the similarity measures and the datasets. Thus, all of these tedious tasks are made transparent to users. Bob can define his own analysis by devising only a few queries on the data lake to start a machine learning process. This centralized architecture of metadata facilitates the work of analysis and saves time for Bob.

5 CONCLUSION & PERSPECTIVES

Without an appropriate metadata management, a data lake can easily turn into a data swamp which is invisible, incomprehensible and inaccessible. Different data lake metadata solutions are proposed with different emphases, however, there is no solution that focuses on analysis metadata which allow users to cross different datasets and analyses to take advantage of the existing datasets information and analyses experience in the data lake.

In this paper, we propose a complete solution of analysis-oriented metadata for data lakes which includes a metadata model, three algorithms of the metadata detection and an application which allow users to search metadata easily. The metadata model contains not only descriptive information on the datasets (statistics on attribute values, management of missing values and relationships between datasets and attributes), but also analytical information on performed analyses of these datasets (studies, tasks and implementations and evaluations of analyses, performances and parameters of the previously executed algorithms. The algorithms explains how to automatically detect analysis-oriented metadata from different structural types of datasets and performed landmarkers or analysis. The application allow users to find information of all the elements stored in the data lake in an ergonomic way. To validate

our solution, we illustrate two usecases with real data to explain the completeness, feasibility and utility of our solution.

For our future work, we plan to include other metadata dedicated to additional types of analyses (other than ML algorithms) such as statistical or OLAP analyses. Moreover, a recommender system may suggest to users the most appropriate algorithms or parameters for different analyses. An explainability system can be considered to explain the results of machine learning analyses as well as other types of analyses. Nevertheless, a massively user-oriented experimentation using our metadata model will be also considered.

REFERENCES

- [1] Qasem Al-Tashi, Said Jadid Abdulkadir, Helmi Md Rais, Seyedali Mirjalili, and Hitham Alhussian. 2020. Approaches to multi-objective feature selection: A systematic literature review. *IEEE Access* 8 (2020), 125076–125096.
- [2] Ayman Alserafi, Alberto Abelló, Oscar Romero, and Toon Calders. 2016. Towards information profiling: data lake content metadata management. In *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)*. IEEE, 178–185.
- [3] Besim Bilalli, Alberto Abelló, Tomas Aluja-Banet, and Robert Wrembel. 2016. Towards Intelligent Data Analysis: The Metadata Challenge. In *IoTBD*. 331–338.
- [4] Alex Bogatu, Alvaro AA Fernandes, Norman W Paton, and Nikolaos Konstantinou. 2020. Dataset Discovery in Data Lakes. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*. IEEE, 709–720.
- [5] Claudia Diamantini, Paolo Lo Giudice, Lorenzo Musarella, Domenico Potena, Emanuele Storti, and Domenico Ursino. 2018. An Approach to Extracting Thematic Views from Highly Heterogeneous Sources of a Data Lake. In *SEBD*.
- [6] Diego Esteves, Agnieszka Lawrynowicz, Pance Panov, Larisa Soldatova, Tommaso Soru, and Joaquin Vanschoren. 2016. ML schema core specification. *W3C*, <http://www.w3.org/2016/10/mls>, Tech. Rep (2016).
- [7] Matthias Feurer, Aaron Klein, Katharina Eggenberger, Jost Springenberg, Manuel Blum, and Frank Hutter. 2015. Efficient and robust automated machine learning. In *Advances in neural information processing systems*. 2962–2970.
- [8] Neil Foshay, Avinandan Mukherjee, and Andrew Taylor. 2007. Does data warehouse end-user metadata add value? *Commun. ACM* 50, 11 (2007), 70–77.
- [9] Alon Y Halevy, Flip Korn, Natalya Fridman Noy, Christopher Olston, Neoklis Polyzotis, Sudip Roy, and Steven Euijong Whang. 2016. Managing Google's data lake: an overview of the Goods system. *IEEE Data Eng. Bull.* 39, 3 (2016), 5–14.
- [10] Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren. 2019. *Automated machine learning: methods, systems, challenges*. Springer Nature.
- [11] Charles M Judd, Gary H McClelland, and Carey S Ryan. 2011. *Data analysis: A model comparison approach*. Routledge.
- [12] C Maria Keet, Agnieszka Lawrynowicz, Claudia d'Amato, Alexandros Kalousis, Phong Nguyen, Raul Palma, Robert Stevens, and Melanie Hilario. 2015. The data mining optimization ontology. *Journal of web semantics* 32 (2015), 43–53.
- [13] Sven Langenecker, Christoph Sturm, Christian Schalles, and Carsten Binnig. 2021. Towards Learned Metadata Extraction for Data Lakes. *BTW 2021* (2021).
- [14] Enrique Leyva, Antonio González, and Raul Perez. 2014. A set of complexity measures designed for applying meta-learning to instance selection. *IEEE Transactions on Knowledge and Data Engineering* 27, 2 (2014), 354–367.
- [15] Rafael G Mantovani, Andre LD Rossi, Edesio Alcobaca, Joaquin Vanschoren, and André CPLF de Carvalho. 2019. A meta-learning recommender system for hyperparameter tuning: Predicting when tuning improves SVM classifiers. *Information Sciences* 501 (2019), 193–221.
- [16] Imen Megdiche, Franck Ravat, and Yan Zhao. 2021. Metadata Management on Data Processing in Data Lakes. In *International Conference on Current Trends in Theory and Practice of Informatics*. Springer, 553–562.
- [17] Cathy O'Neil and Rachel Schutt. 2013. *Doing data science: Straight talk from the frontline*. "O'Reilly Media, Inc".
- [18] Panče Panov, Larisa Soldatova, and Sašo Džeroski. 2014. Ontology of core data mining entities. *Data Mining and Knowledge Discovery* 28, 5-6 (2014), 1222–1265.
- [19] Franck Ravat and Yan Zhao. 2019. Data lakes: Trends and perspectives. In *International Conference on Database and Expert Systems Applications*. Springer, 304–313.
- [20] Franck Ravat and Yan Zhao. 2019. Metadata management for data lakes. In *European Conf. on Advances in Databases and Information Systems*. Springer, 37–44.
- [21] William Raynaut, Chantal Soule-Dupuy, and Nathalie Valles-Parlangeau. 2016. Meta-Mining Evaluation Framework: A large scale proof of concept on Meta-Learning. In *Australasian Joint Conference on Artificial Intelligence*. Springer, 215–228.
- [22] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why should I trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 1135–1144.
- [23] Adriano Rivoli, Luís PF Garcia, Carlos Soares, Joaquin Vanschoren, and André CPLF de Carvalho. 2018. Towards reproducible empirical research in meta-learning. *arXiv preprint arXiv:1808.10406* (2018), 32–52.
- [24] Tyler J Skluzacek, Rohan Kumar, Ryan Chard, Galen Harrison, Paul Beckman, Kyle Chard, and Ian Foster. 2018. Skluma: An extensible metadata extraction pipeline for disorganized data. In *2018 IEEE 14th International Conference on e-Science (e-Science)*. IEEE, 256–266.
- [25] Saúl Solorio-Fernández, J Ariel Carrasco-Ochoa, and José Fco Martínez-Trinidad. 2020. A review of unsupervised feature selection methods. *Art. Intel. Review* 53, 2 (2020), 907–948.
- [26] Jovan Varga, Oscar Romero, Torben Bach Pedersen, and Christian Thomsen. 2014. Towards next generation BI systems: The analytical metadata challenge. In *International conference on data warehousing and knowledge discovery*. Springer, 89–101.
- [27] Katarzyna Woźnica and Przemysław Biecek. 2020. Towards better understanding of meta-features contributions. *arXiv preprint arXiv:2002.04276* (2020).
- [28] Yi Zhang and Zachary G Ives. 2020. Finding Related Tables in Data Lakes for Interactive Data Science. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. 1951–1966.

Priority-Based Skyline Query Processing for Incomplete Data

Chuang-Ming Liu
Dept Comp Sci & Informat Engn
National Taipei University of
Technology
Taipei, Taiwan(R.O.C)
cmliu@csie.ntut.edu.tw

Denis Pak
Technical Research Department
Infotrend Technology
Taipei, Taiwan(R.O.C)
rikaho@mail.ru

Ari Ernesto Ortiz Castellanos
Coll Elect Engn & Comp Sci
National Taipei University of
Technology
Taipei, Taiwan(R.O.C)
t107999401@ntut.org.tw

ABSTRACT

Over the years, several skyline query techniques have been introduced to handle incompleteness of data, the most recent of which has proposed to sort the points of a dataset into several distinct lists based on each dimension. The points would be accessed based on these lists in round robin fashion, and the points that haven't been dominated by the end would compose the final skyline. The work is based on the assumption that relatively dominant points, if sorted, would be processed first, and even if the point wouldn't be a skyline point, it would prune huge amount of data. However, that approach doesn't take into consideration that the dominance of a point depends not only on the highest value of a given dimension, but also on the number of complete dimensions a point has. Hence, we propose a Priority-First Sort-Based Incomplete Data Skyline (PFSIDS) that utilizes a different indexing technique that allows optimization of access based on both number of complete dimensions a point has as well as sorting of the data.

CCS CONCEPTS

• Information systems → Spatial-temporal systems;

KEYWORDS

Skyline, Incomplete Data, Query Processing, Spatial-temporal Data, Data Management

ACM Reference Format:

Chuang-Ming Liu, Denis Pak, and Ari Ernesto Ortiz Castellanos. 2021. Priority-Based Skyline Query Processing for Incomplete Data. In *25th International Database Engineering & Applications Symposium (IDEAS 2021)*, July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 8 pages. <https://doi.org/10.1145/3472163.3472272>

1 INTRODUCTION

The skyline query is a query that results in a “skyline” – a set of the most interesting, or best, points from the whole dataset[2][3][4][5]. The “best” points are the points that are not dominated by any other points, i.e. a point dominates another point when it is not worse in all dimensions and is better in at least one. A scatter plot

of a two-dimensional dataset of points that represent the prices of hotel rooms with varied prices and distances to the beach. The points dominated by the skyline points are located inside the region allocated by colored lines of their respective points. For example, *A* dominates *D* and *E* by having both lower prices and lower distances to the beach, similarly, *B* dominates *C*, *E*, *F*, and *J*, etc. When points *A*, *B*, and *G* are compared, we would notice that they don't dominate each other since *A* has the lowest distance to the beach and hence cannot be dominated based on this dimension, *G* has the lowest price, and *B* has a good combination of both, which allows it not to be dominated by *A* and *G*, since it has a better price than *A* and better distance to the beach than *G* (see figure 1).

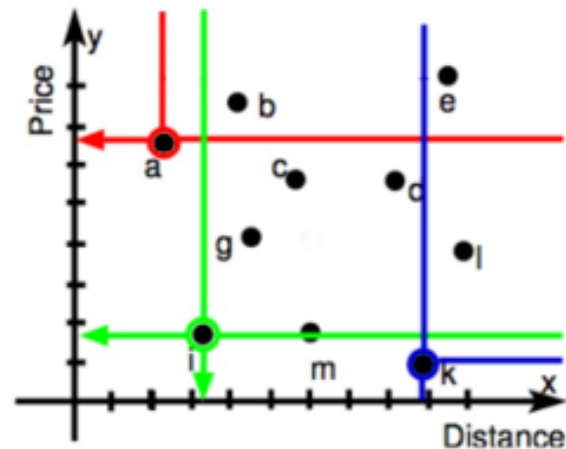


Figure 1: Skyline points *a*, *i*, and *k* in the illustrated example

Most of the real-world data-driven applications have to deal with data that is incomplete due to factors as sensor failures, obstruction of signal, measurement errors, and others. Those factors make a problem for setting the skyline requirement for the incompleteness of data[6][7][8]. Finally, the other sections of this work are organized as follows. Sorted-Based Incomplete Data Skyline Algorithm in section 2, our contributions in section 3, our proposed approach in section 4, the experiments in section 5 and finally our conclusions in section 6.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](https://www.acm.org).

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Association for Computing Machinery.

ACM ISBN 978-1-4503-8991-4/21/07...\$15.00

<https://doi.org/10.1145/3472163.3472272>

2 SORT-BASED INCOMPLETE DATA SKYLINE ALGORITHM

Sort-based Incomplete Data Skyline algorithm (SIDS), where the algorithm creates d sorted *lists* based on each dimension. For example a sorted lists based on the movie ratings dataset, where users are considered as dimensions. For example(See figure 2), the first dimension, u_1 , consists of movies 5, 4, 1 and 7 in that exact order, since they are sorted on based on their values in this dimension only. Then, it chooses one of the lists as a starting point in a round-robin fashion. Each point of the list would be compared to each other, pruning away all of the dominated tuples. Then, the algorithm moves to another list. If the algorithm reaches the last list, the pointer to the current position is increased, and the current list is changed to the first one. If the tuple has been processed k times, where k is the number of complete dimensions, and was not dominated by any other point, then it is determined to be a skyline point. Such an approach has several distinct disadvantages: 1) It doesn't take into account the distinct feature of incomplete data: the relative power of a point, in other words the overall number of points it may dominate, depends not only on the individual values in any given dimension but also on the number of complete points it has.. 2) Due to the nature of the algorithm, addition of the new data is very costly, as it would require to remake and resort all of the lists.

u_1	u_2	u_3	u_4	u_5
m_5	m_7	m_1	m_5	m_3
m_4	m_2	m_2	m_2	m_5
m_1	m_6	m_5	m_7	m_4
m_7		m_7	m_4	m_7
		m_3	m_1	

Figure 2: Sorted lists in SIDS

3 CONTRIBUTIONS

3.1 Effects of the number of complete dimensions on performance

The primary contribution of this work is the exploration of the observed improvement of performance based on prioritization of points with low number of complete dimensions. To the best of our knowledge, this notion has not been discussed before.

3.2 Addressing the limitation of related works

In this work we explain and improve on some limitations present in related works.

3.3 Skyline Query in Incomplete Data

Another contribution of this work is the development of an algorithm that takes advantage of the usage of both values in a given dimension and number of dimensions a point has in order to improve the overall performance of skyline query in incomplete data.

4 PROPOSED APPROACH

Priority-First Sort-based Incomplete Data Skyline (PFSIDS) utilizes a different indexing technique that allows optimization of access based on both a number of complete dimensions a point has as well sorting of the data. This approach has demonstrated evaluation efficiency and scalability in datasets on synthetic and real-world.

4.1 Preliminaries

We assume that dataset D is a d dimensional dataset with dimensions $d = d_1, d_2, d_3, d_4, \dots, d_n$. Each point $p \in D$ is represented by a set of values $p = p_1, p_2, p_3, p_4, \dots, p_n$ (see figure 3). The incompleteness of data is represented by missing values in each dimension present in the dataset, where missing data is denoted as -. For example, a point p in a five-dimensional dataset with values a, b, c, d in the first four dimensions and a missing value in the last dimension would be represented as $(a, b, c, -,)$ in figure 4. Without loss of generality, this work assumes that all dimensions of the dataset have a total order, in other words, greater values are considered to be better.

4.1.1 Creating Index. This step adapts the dataset into an index that will be used for the processing of the skyline. The pseudo-code is shown in Algorithm 1. In lines 1-2 a list is created for each combination of a cumulative number of non-missing dimensions observed inside of the dataset. For example, the dataset shown simple table can show the points with cumulative numbers of complete dimensions of 1, 2, 3, 4 and 5. After which, in line 3, all of the points with a corresponding number of complete dimensions are initialized into variable `listPoints`. Then, in lines 4-9 all of the points in `listPoints` are sorted into several arrays (see Algorithm 1).

All the list would be arranged in increasing order of number of dimensions, representing a "priority" of access to the points, since as mentioned previously the less dimensions a point has, the more points it would subsequently prune (See table 2(a), 2(b), 2(c), 2(d) and 2 (e)) in figure 4.

Algorithm 1 consists of the creation of a list, it receives first a dataset denoted D . The loop is the representation of a different cumulative number of complete dimensions of the points, in this case, all of them. We declare an array called `which` content our list, then we initialize the `linePoints` when at that moment the loop counter (i) is equal to parameters. Subsequently, we use a second array to attach the dimensions from our dataset denoted D . Inside the second loop we initialize the array and start to sort all points which belong to our line points and stored them in the new list. When both loops complete the number of line points of the dataset, we receive an array with all lists in ascending order.

4.1.2 Deriving Skylines. This stage takes in the index created in the previous stage and produces the skyline of a dataset[9][10][11]. Algorithm 2 represents the pseudo-code of the procedure. First, in line 1 we initialize *CandidateSet* to be equal to the whole dataset and the initial Skyline to be *null*, as at the beginning we have to consider the whole dataset to be a potential skyline and the skyline itself should be empty because no points have been processed yet. Since points with multiple numbers of complete dimensions are repeatedly encountered inside the arrays of the index, a structure

Point	d ₁	d ₂	d ₃	d ₄	d ₅
p1	10	8	3	7	4
p2	9	-	-	-	-
p3	-	7	5	2	5
p4	7	-	-	8	-
p5	3	7	1	6	2
p6	8	-	7	-	-
p7	-	-	2	6	3
p8	4	4	4	5	8
p9	1	3	-	1	-
p10	-	-	-	-	3
p11	2	9	8	-	-
p12	5	-	-	-	-
p13	-	3	2	8	3
p14	-	-	-	5	-
p15	-	3	-	-	-
p16	-	-	-	-	-
p17	-	9	-	-	-
p18	-	-	7	-	-
p19	7	-	4	6	7
p20	8	5	7	3	2

Figure 3: Sample of Dataset

has to be introduced in order to decrease redundancy. Lines 2-5 represent the initialization of *processedCount* and *dimCount* variables for each point of the dataset. The first one keeps track of how many times a point has been processed and the second one represents the cumulative number of complete dimensions a point has. Thus, if the point has been processed the same amount of time as the number of complete dimensions it has and has not been dominated by any other points it could be considered as a part of the skyline. Then, the index is processed in a round-robin fashion, where a list, corresponding to the number of complete dimensions is accessed based on the order of priority, the points are iteratively processed based on the position pointer. When the last dimension of a given list is processed, the algorithms continue to the next list. After the last list is processed, the position pointer is increased by 1. If the value is missing in an array of the index, the algorithm simply skips the dimension. If we take the index from Figure 7 as

1 Complete Dim.					2 Complete Dim.					3 Complete Dim.				
d ₁	d ₂	d ₃	d ₄	d ₅	d ₁	d ₂	d ₃	d ₄	d ₅	d ₁	d ₂	d ₃	d ₄	d ₅
p2	p17	p18	p14	p10	p6	-	p6	p4	-	p11	p11	p11	p7	p7
p12	p15	-	-	-	p4	-	-	-	-	p9	p9	p7	p9	-

4 Complete Dim.					5 Complete Dim.				
d ₁	d ₂	d ₃	d ₄	d ₅	d ₁	d ₂	d ₃	d ₄	d ₅
p19	p3	p3	p13	p19	P1	P1	P20	P1	p8
-	p13	p19	p19	p3	P20	p5	p8	p5	P1
-	-	p13	p3	p13	p8	p20	p1	p8	p5
					p5	p8	p5	p20	p20

Figure 4: An index example of the sample dataset

Algorithm 1: Creating lists	
Input: dataset D	
Output: lists l that contain sorted arrays for each dimension r_{di}	
for each different cumulative number of complete dimensions across all of the points i do	
create a list l_i	
initialize $listPoints = p \in D$ where $ p == i$	
for each dimension $d \in D$ do	
create an array r_{di}	
sort all points $p \in listPoints$ based on d	
store points in r_{di}	
insert r_{di} into l_i	
end	
end	
sort all lists l in an ascending order of i	
return l	

Figure 5: Algorithm 1

an example, first the list corresponding to points with 1 complete dimension is accessed and the position pointer is equal to 0. Then, the points $p2, p17, p18, p14, p10$ are processed. After that, the procedure continues to the next list, where $p6$ is processed twice and $p4$ once. When the last list is finished, the position pointer is increased by one, and the operation advances in a similar fashion until the *CandidateSet* of points is empty. Meaning that all of the points inside the *CandidateSet* are either pruned away or promoted into the Skyline.

Initially, each point would set a flag variable is dominated to *False*, as indicated in line 11. Then, if *processedCount* of the point is equal to 0 the point would be compared against all other points in *CandidateSet*. If the points inside the *CandidateSet* are dominated, they would be removed from the set. On the other hand, if the point is dominated by any of the candidates, its flag variable *isDominated* is set to *True*. The point isn't removed right away because while it

is dominated, the point itself can still prune other points. By the end of exhaustive comparisons, line 21 checks if p is both inside the *CandidateSet* and *isDominated* and removes it if that is true. Lines 25-28 increase the *processedCount* by 1, and if the variable reaches the cumulative number of complete dimensions *dimCount* the point has while still being inside the *CandidateSet*, the point is moved from the set into Skyline (See figure 6).

In Algorithm 2, the list generated from Algorithm 1 is implemented, the initial values are set based on Candidates equivalent as Dataset complete. Skyline is null at that moment and we apply an iterator for our loop. We can say that each point in our loop is equivalent to our dataset. Inside of loop we set the current position and set a flag variable as false. The reason is to generate an extraction of positions, next we verify if the position is dominant if it is not true this register is going to be removed from *CandidateSet*, if the case is opposite it will be considered as dominant, now we have the point and its state, if dominant then the point is removed from *CandidateSet*, we evaluate the current state versus next one, for trying to avoid redundancy. The process is repeated, again and again, and the state dominant (true or false) indicates which elements must be move to the next array Skyline.

4.2 CandidateSet data structure

It is important to note that throughout the runtime of the algorithm *CandidateSet* is accessed very frequently for both lookups and removals of the candidates themselves[12][13][14]. And as such, it is very important to select an appropriate data structure so that the algorithm does not get bottlenecked by it. Therefore a hash table was chosen, as it has an amortized performance of $O(1)$ for both search and deletion.

4.3 Redundant comparison minimization.

Since all of the points in the dataset would have to be processed due to the cyclic dominance[15][16][17] it is critical for the performance of the algorithm to avoid comparisons that have already been done. Previous approaches proposed to use Timestamps, where all of the points would have their corresponding variable that would hold the time when it was last processed. However, to reduce the amount of memory used we propose to utilize the index that has been already created. The idea behind it is very simple: if a point is located in a previous list with an equal or higher position, the points have been already compared before[18][19][20][21].

4.4 Discussion on time and space complexity

The worst case time complexity of PFSISD is $O(dn^2 + dn \log n)$ which would result in final $O(n^2)$. The same worst-case time complexity is observed in all skyline query techniques in incomplete data, since if the query fails to prune away the points, it would require to do exhaustive pairwise comparisons.

The best case is $O(dn + dn \log n)$ which results in $O(n \log n)$. That is worse best-case time complexity than the one observed in the naïve approach since it doesn't sort its points in any way. However, it is highly unlikely to have anywhere near this performance in a real-world scenario.

Algorithm 2: Skyline Query Processing

```

Input: lists  $l$  that contain sorted arrays for each dimension  $r_d$ 
Output: Skyline set  $S$ 
  initialize CandidateSet =  $D$ , Skyline =  $\emptyset$ , iteration = 0
  for each point  $p \in \text{CandidateSet}$  do
    initialize processedCount( $p$ ) = 0
    initialize dimCount( $p$ ) = number of complete dimensions in  $p$ 
  end
  while CandidateSet  $\neq \emptyset$  do
    for each list  $l_i$  do
      for each complete dimension  $d \in l_i$  do
         $p = r_d[\text{position}]$ 
        isDominated = False
        if processedCount( $p$ ) = 0 then
          for each candidate  $c \in \text{CandidateSet}$  do
            if  $p$  dominates  $c$  then
              remove  $c$  from CandidateSet
            else if  $c$  dominates  $p$  then
              isDominated = True
            end
          end
        end
        if  $p \in \text{CandidateSet}$  and isDominated then
          remove  $p$  from CandidateSet
        end

        processedCount( $p$ ) = processedCount( $p$ ) + 1
        if  $p \in \text{CandidateSet}$  and processedCount( $p$ ) = dimCount( $p$ ) then
          move  $p$  from CandidateSet to Skyline
        end
        iteration = iteration + 1
      end
    end
  end
  return Skyline

```

Figure 6: Algorithm 2

Calculating the average case is very tricky, as it would largely depend on the effectiveness of preprocessing. In our experience, datasets with anti-correlated and random distributions would exhibit time complexity very close to $O(n \log n)$.

The memory required by SIDS[1] is higher because in addition to the dataset and preprocessed data it has to also store the TimesStamps required for minimization of redundant comparisons, whereas our approach utilizes the preprocessed data and therefore has no need for the TimesStamps.

5 EXPERIMENTS

Experiments have been carried out on both real-world and synthetic datasets for a more precise evaluation of the performance. Real-world datasets are represented by NBA dataset as an example of correlated distribution and the New York City Airbnb Open dataset (see figure 7, 8, 9, and 10), since mention that hotel data can be seen as an example of anti-correlated distribution. The synthetic

dataset is generated by the NumPy python library with random distribution.

NBA is a dataset that contains aggregate individual player statistics from 67 regular seasons. The dataset represents almost 4000 players, with 47 recorded over the years. The dataset is positively correlated, which means that people with high skills in some fields tend to be also highly skilled in other areas. The missing rate of the dataset is 23 percentage.

New York City Airbnb Open dataset contains a summary of information and metrics for listings in New York City. It contains 47906 tuples with 6 dimensions that recorded price, number of ratings, etc. The missing rate of the dataset is 11 percent.

Several synthetic datasets have been utilized for extensive testing. All of them have varying sizes, a number of dimensions, and missing rates. For fairness experiments on datasets have been carried out on 20 different seeds, after which the results have been averaged out.

Processing time and a number of comparisons have been adopted as metrics for the evaluation of scalability, as well as the effect dimensionality and missing data have on the performance. The standard missing rate for evaluation of scalability and dimensionality is 20 percent, the same missing rate was used in previous works. The preprocessing time required for the algorithms, unless specified otherwise, has been included in the results.

5.1 Scalability

Figure 9 shows that PFSIDS consistently reduces the overall number of comparisons required for determining the final skyline. The reason being that the algorithm prioritizes the points that are more likely to prune the most amount of points. The reduction of a number of comparisons in figures (8), (10), and (12) leads to improvement in processing time, as shown in (7), (9), and (11). The low difference in the number of comparisons for the NBA dataset is caused by the correlated distribution of data, where the more dominant points are located at the top of the lists.

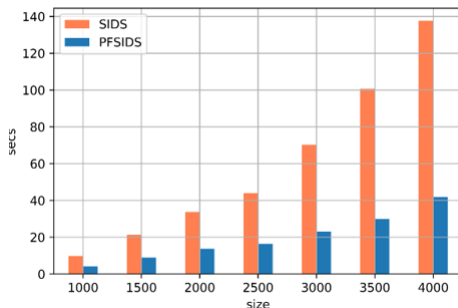


Figure 7: NBA – Processing Time

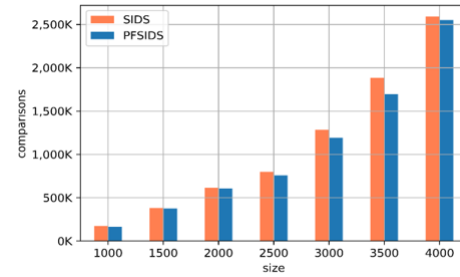


Figure 8: NBA-Number of Comparisons

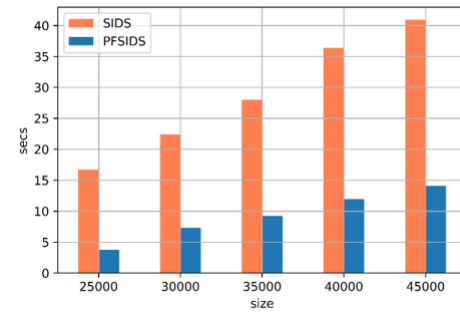


Figure 9: NYC Airbnb – Processing Time

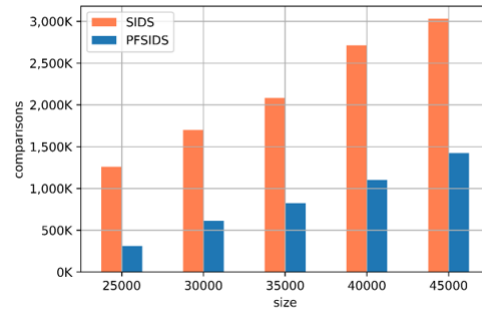


Figure 10: (NYC Airbnb Comparisons)

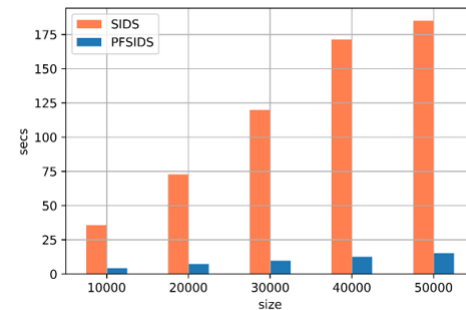


Figure 11: Synthetic – Processing Time

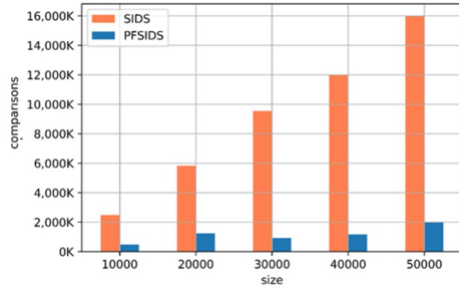


Figure 12: Synthetic – Comparisons

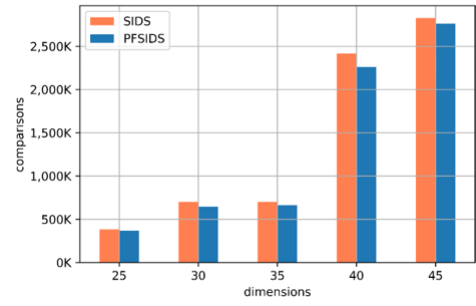


Figure 14: NBA – Number of comparisons

5.2 Dimensionality

As shown in figures 13, 14, 15, 16, 17, and 18 PFSIDS outperforms all other approaches. Here, similar to the scalability of the size benchmark, we can see that PFSIDS outperforms SIDS across the board. NBA dataset behaves in a similar fashion to the one we see in Figures 13 and 14. The NYC dataset, however, performs similarly to our approach up until we reach 4 dimensions. Such behavior is caused by the fact that not all of the dimensions of the NYC dataset are anti-correlated. The first four dimensions have correlated distribution according to each other, and when anti-correlated dimensions are processed in later stages, the performance levels out to the one similar to the one observed in figures 15 and 16.

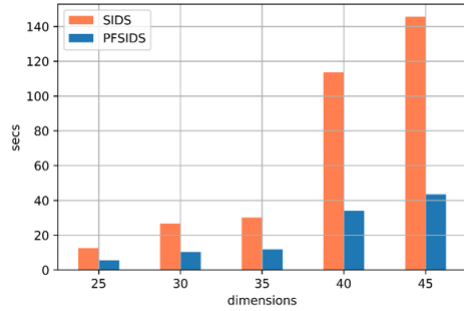


Figure 13: NBA – Processing Time

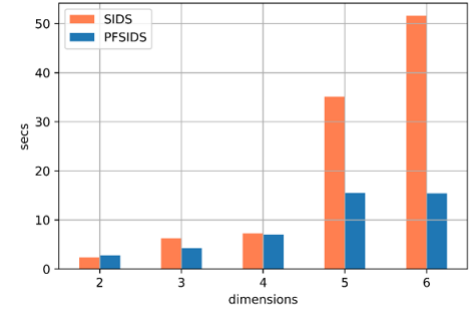


Figure 15: NYC – Processing Time

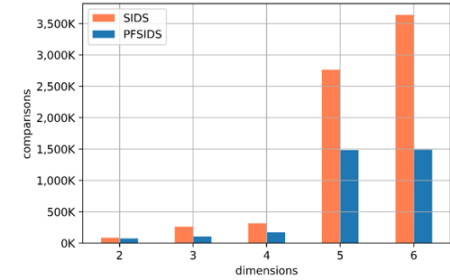


Figure 16: NYC – Number of comparisons

5.3 Missing rate

Since all of the non-synthetic datasets have a static missing rate of data, 23 percentage for NBA and 9 percentage for NY Airbnb, only synthetic data has been used for this benchmark, as the missing rate can be easily adjusted. Figures 19 and 20 show that the difference in the processing time of PFSIDS and SIDS is beginning to become rather negligible with a missing rate greater than 70 percent. That is because at that point the construction of the index starts to dominate the performance of the algorithm, while SIDS starts accessing more desirable points with a small number of dimensions earlier because of the high missing rate of the data.

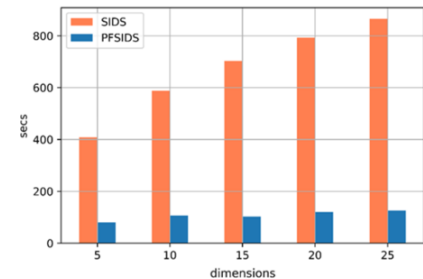


Figure 17: Synthetic – Processing Time

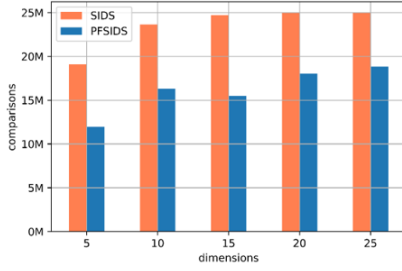


Figure 18: Synthetic – Number of comparisons

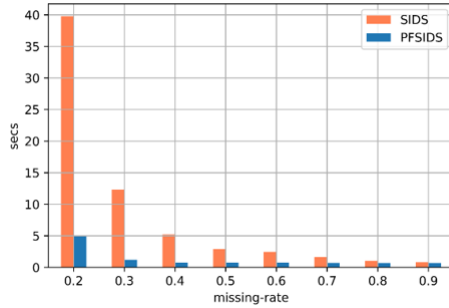


Figure 19: Synthetic – Processing time

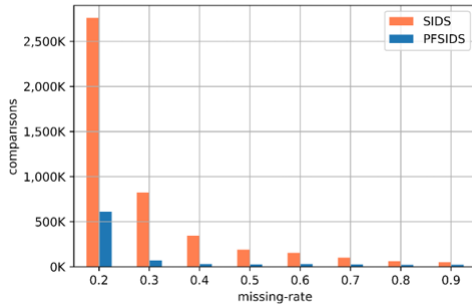


Figure 20: Synthetic – Number of comparisons

5.4 Pre-processing

As mentioned before, one disadvantage of our approach is that it requires pre-processing of the data, and while pre-processing of the data before the query is feasible in most real-world applications, it may still pose a concern with larger datasets, where it starts to dominate the performance of the query. And as such we have performed several tests to evaluate how pre-processing of data scales with different amounts of data, number of dimensions, and missing rates, as well as how it affects the performance of the whole algorithm. As can be observed in figures 21, 22, 23 the number of dimensions and missing rate have no observable impact on the performance of PFSIDS, whereas the size of the dataset linearly affects the time it takes to create the index.

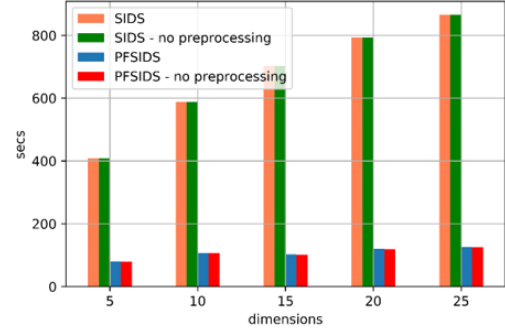


Figure 21: Effect of size

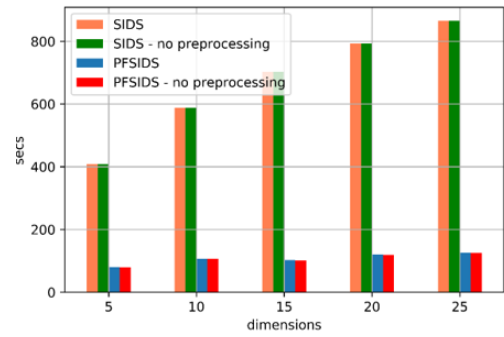


Figure 22: Dimensionality

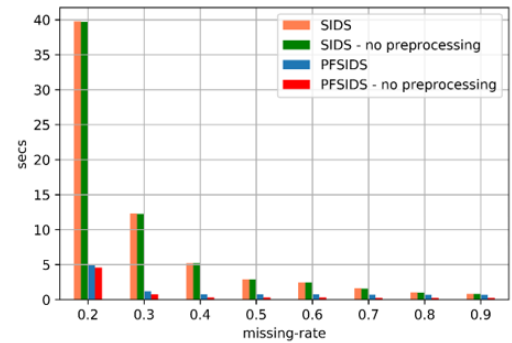


Figure 23: Missing rate

5.5 Meaningful Skylines

We have decided to remove the points with a small number of complete dimensions, which would theoretically give us more meaningful skylines, as the points with a small number of complete dimensions may not be very desirable as a result. The x-axis of figures 24 and 25 represent the points with less than or an equal number of complete dimensions that are removed from the dataset. We can observe that the removal of points with 1 complete dimension has a considerable effect on the processing time and number of comparisons for PFSIDS, after which it levels out. The SIDS algorithm seems to be unaffected by the removal because it does not

depend on the points with a low number of complete dimensions. Nevertheless, PFSIDS exhibits a better performance overall.

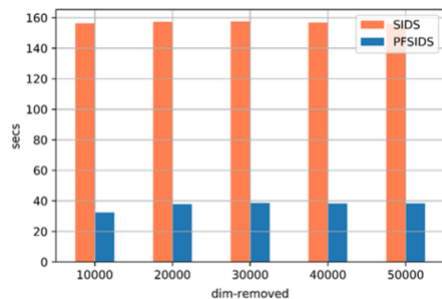


Figure 24: Effect of removal of points with low number of complete dimensions

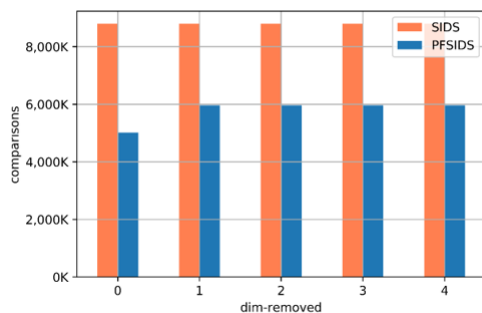


Figure 25: Effect of removal of points with low number of complete dimensions

6 CONCLUSIONS

In this work, we have proposed a new approach for processing skyline queries in incomplete data, PFSIDS. We utilize the observation that the relative dominance of a point depends not only on the individual values the points have in a given dimension but also on the number of complete dimensions a point has. In PFSIDS, we create a specialized index that allows us to prioritize points in the dataset that are more likely to dominate a large number of points, which results in performance improvement. By comparing our approach with existing works, we observed that PFSIDS consistently reduces the number of overall comparisons which shortens the processing time considerably. The experimental results show our approach can achieve a performance improvement of up to several orders of magnitude for missing rates less than 70 percent.

REFERENCES

- [1] R. Bharuka and P. S. Kumar. "Finding Skylines for Incomplete Data". *Proceedings of the Twenty-Fourth Australasian Database Conference*, vol. 137, pp.109-117, January 23, 2013.
- [2] Q. Ma, Y. Gu, W. Lee and G. Yu. "Order-Sensitive Imputation for Clustered Missing Values". *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 1, pp. 166-180, January 1, 2019
- [3] X. Miao, Y. Gao, S. Guo and W. Liu. "Incomplete data management a survey". *Frontiers of Computer Science*, Springer. April 3, 2018.
- [4] Y. Wang, Z. Shi, J.L. Wang, L.F. Sun, B.Y. song. "Skyline Preference Query Based on Massive and Incomplete Dataset". *Special Section on Big Data Analytics in Internet of Things and Cyber-Physical Systems*, February 17, 2017.
- [5] X. Miao, Y. Gao, S. Guo, L. Chen, J. Yin and Q. Li. "Answering Skyline Queries over Incomplete Data with Crowdsourcing". *IEEE Transactions on Knowledge and Data Engineering*, October 1, 2019.
- [6] Y. Gulzar, A.A. Alwan, S. Turaev, "Optimizing Skyline Query Processing in Incomplete Data". *IEEE Access*, December 6, 2019.
- [7] Y. Gulzar, A. A.Alwan, A. Z. Abualkashik, A. Mehmood, "A Model for Computing Skyline Data Items in Cloud Incomplete Databases". *Procedia Computer Science*, April 6-9, 2020.
- [8] Y. Gulzar, A. A. Alwan, N. Salleh, A. Shaikhli, "Identifying Skylines in Cloud Databases with Incomplete Data". *Journal of Information and Communication Technology*, December 20, 2019.
- [9] , Y. Sidi and O. Harel, "The treatment of incomplete data: Reporting, analysis, reproducibility, and replicability". *Social Science and Medicine*, May 11, 2018.
- [10] A. amarilli, M. L. Ba, D. deutch and P. Senellart, "Computing possible and certain answers over order-incomplete data." *Theoretical Computer Science*.
- [11] S.H. Willemse, L. H. E Karssemakers, M.A. E. M Oomens, W.H Schreuder, J. A. Lindeboom, A. J. van Wijk and J. de Lange, "Cervicofacial non-tuberculous mycobacterial lymphadenitis clinical determinants of incomplete surgical removal". *International Journal of Oral and Maxillofacial Surgery*, December 6, 2020.
- [12] J.J. Gu, Z.B. Jiang, Y.S. Sun, M. Zhou, S. M. H. Liao and J. J. Chen, "Spatio temporal trajectory estimation based on incomplete Wi-Fi probe data in urban rail transit network". *Journal Pre-Proof*, January 9, 2020.
- [13] W. Xiang and W. Zhou. "Bayesian network model for predicting of third-party damage to underground pipelines and learning model parameters from incomplete datasets", . *emphReliability Engineering and System Safety*, October 1, 2020.
- [14] , J. H. Kim and B. S. Ahn, "Extender VIKOR method using incomplete criteria weights" . *Exper Systems With Applications*, November 11, 2019.
- [15] N. N. Thuy and S. Wongthanavasu, "An efficient stripped cover-based accelerator for reduction of attributes in incomplete decision tables", . *Expert Systems with applications*, April 1, 2019.
- [16] C. Guo, W. k . H, F. Yang and D. X. Huang, "Deep learning technique for process fault detection and diagnosis in the presence of incomplete data", . *Chinese Journal of Chemical Engineering*, June 11, 2020.
- [17] B. J Santoso and T.Y. Connery, "Answering Why Not Questions on Reverse Skyline Queries Over Incomplete Data", . *JUTI: Journal Ilmiah Teknologi Informasi*, January 4, 2019.
- [18] A. Alwan, H.Ibrahim, N. Udzir and F. Sidi, "Missing Values Estimation for Skylines in Incomplete Database", . *The International Arab Journal of Information Technology*, January 1, 2018.
- [19] K. Q. Zhang, H. Gao, X. X. Han, Z.P. Cai and J. Z. Li, "Modeling and Computing Probabilistic Skyline on Incomplete Data", . *emphIEEE Transactions on Knowledge and Data Engineering*, March 3, 2020.
- [20] Y. X. Hsn and Z. C He, "Simultaneous Incomplete Traffic Data Imputation and Sililar Patter Discovery with Bayesian Nonparametric Tensor Decomposition", . *Journal of Advanced Transportation*, July 20, 2020.
- [21] X. F Zhu, J. Y. Yang, C. Y Zhang and S. C Zhang, "Effient Utilization of Missing Data in Cost Sensitive Learning", . *Transactions of Knowledge and Data Engineering*, Novemver 28, 2019.

Looking for Jobs? Matching Adults with Autism with Potential Employers for Job Opportunities

Joseph Bills

Computer Science Dept., Brigham Young University
Provo, Utah, USA
roboiguana@gmail.com

Yiu-Kai Ng

Computer Science Dept., Brigham Young University
Provo, Utah, USA
ng@compsci.byu.edu

ABSTRACT

Adults with autism face many difficulties when finding employment, such as struggling with interviews and needing accommodating environments for sensory issues. Autistic adults, however, also have unique skills to contribute to the workplace that companies have recently started to seek after, such as loyalty, close attention to detail, and trustworthiness. To work around these difficulties and help companies find the talent they are looking for we have developed a job-matching system. Our system is based around the stable matching of the Gale-Shapley algorithm to match autistic adults with employers after estimating how both adults with autism and employers would rank the other group. The system also uses filtering to approximate a stable matching even with a changing pool of users and employers, meaning the results are resistant to change as the result of competition. Such a system would be of benefit to both adults with autism and employers and would advance knowledge in recommender systems that match two parties.

CCS CONCEPTS

• **Information systems** → **Information retrieval**; • **Retrieval tasks and goals** → *Recommender systems*.

KEYWORDS

Autism, job search, stable mapping, algorithms, recommendations

ACM Reference Format:

Joseph Bills and Yiu-Kai Ng. 2021. Looking for Jobs? Matching Adults with Autism with Potential Employers for Job Opportunities. In *25th International Database Engineering & Applications Symposium (IDEAS 2021)*, July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3472163.3472270>

1 INTRODUCTION

Adults with autism are among the most under employed demographics, with a recent report claiming that 85% of them are unemployed [15]. This statistic, however, is not due to an inherent lack of ability of adults with autism, as is proven by the fact that some intervention can improve rates of employment. Assisting adults

with autism find employment provides benefits both to the individual and to society. For the individual, gainful employment leads to financial independence which in turn leads to increased opportunities for adults with autism. Meaningful employment also leads to increased self-esteem and general well-being, even leading to increased cognitive ability [9]. For society, increased independence also results in less social expenditure and employment increases tax revenue [10]. More importantly although individuals with autism have special skills to contribute to the workplace if applied to the right job, this talent is currently not being utilized. It is for this reason that there is an urgent need to find a solution to this problem.

One technique that has proven to be successful in improving rates of employment and retention is job matching [4]. Resources for job matching for adults with autism, however, are limited. The problem that we tackle is to develop an algorithm to provide an automated job-matching system for adults with autism and potential employers who are interested in hiring adults with autism. Job matching has proven to be a successful technique in helping adults with autism find employment and remain employed. Existing programs that utilize job matching for adults with autism include Swedish corporation *Samhall* (samhall.se) and American corporation *Daivergent* (daivergent.com). *Samhall*'s system includes matching client's abilities with employer's demands using a system that measures 25 different traits (covering sensory function, intellectual ability, mental ability, social ability, and physical ability) on 3 levels (limited, good, and high ability on the client side corresponding to low, medium, and high requirements on the employer side) [16]. *Daivergent* uses artificial intelligence to match vetted candidates with jobs that they extracted from descriptions using machine learning [3]. Unfortunately, there is little public information about how these corporations provide their matching beyond what details they choose to share with the public, both limiting their services to their clientele and restricting potential research contributions from studying their systems.

To solve this problem, we develop a *job-matching* system so that both users, i.e., adults with autism, who look for work, and employers can autonomously create profiles and then be automatically matched. Like *Samhall*, this matching is done by quantifying the skills of employees and demands of work on multiple axes. The goal is to pick a match that *minimizes* the discrepancy between the user's skills and the employer's demands as this will minimize the amount of skills the user would need to develop and accommodations that the employer would have to make. In addition to measuring the job tasks itself, demands for the application process such as required interview skills are also included as part of the work demands.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Association for Computing Machinery.

ACM ISBN 978-1-4503-8991-4/21/07...\$15.00

<https://doi.org/10.1145/3472163.3472270>

Our job-matching system takes into account not only the *skills* of the user, but also their *interests*. This not only respects the desires of the individual, but also leads to much greater productivity [13]. As we assume that the user’s preference is based principally on their *interest* while an employer’s preference is based principally on the ability for a worker to perform the task *effectively* according to their *skills*, *interest* aspects are measured separately from *skill* aspects. The fact that the user’s preference and the employer’s preference can differ significantly necessitates a system for finding a compromise between the two. From the *interest* aspects, an automated ranking of employers is generated for each user, and from the *skill* aspects, an automated ranking of users is generated. These different rankings are combined into a single match using the Gale-Shapley stable algorithm [6], which finds a *stable match*. **Stable** means that no two participants may both have a higher ranked match with each other than who they were already paired with in the stable match.

The proposed job-matching system advances both technology for assisting adults with autism in finding jobs and the knowledge of matching systems in general. The fact users act autonomously in this system means that the labor costs associated with current job-matching systems can be reduced. Positions are also extremely limited in existing job-matching systems that are geared towards adults with autism, so this system acts as a potential starting point for increasing access to many more adults with autism. It can potentially be extended to serve other populations as well. Moreover, our job-matching system differs from existing systems in that it is open to use for anyone who wishes, no manual vetting is required. It is also open source, so it may be built upon for further research.

2 RELATED WORK

While numerous companies use their own job-matching algorithms, academic research on the subject exists as a specific application in the broader field of recommendation algorithms. CASPER [14], one of the proposed job-recommendation algorithms, focuses on clustering users based on their activities while reviewing jobs so that collaborative filtering may be applied. Malinowski et al. [12], on the other hand, use the content-based filtering approach based on profiles that are manually entered by users. Much of the latest research works on job-matching relate to processing data from resumes and other sources so it can be used, with Resumatcher introduced by Guo et al. [8] in particular who seek to match similar profiles based on extracting data from unstructured resumes and job descriptions. Others, which focus on comparing unstructured data so that similar profiles may be matched, include the self-reinforcing model proposed by Koh et al. [11] and the collective-learning approach developed by Cing [2].

Even though research works on algorithms for matching autistic adults with employers is minimal, there are substantial works on the subject of autistic employment in general. Of particular interest is the work of Dreaver et al. [4] who tackle the problem from an employer’s perspective. In addition to suggesting matching, they emphasize the importance of external supports and employers understanding autism. Indeed, most research works focus on the perspective adults, with numerous studies supporting the efficiency of Behavior Skills Training (BST), especially when combined with prompting and audio cues. Grob et al. [7] managed to

achieve a 100% success rate at teaching skills using BST combined with prompting, while Burke et al. [1] found that BST combined with audio cues is six times as effective as BST by itself.

We observe that although existing job-recommender algorithms cannot assign multiple users to the same job, existing job-matching strategies can. Moreover, existing job-matching approaches attempt to automate the recruitment stage in the hiring process [2], which aims to make suggestions to many interested job seekers instead of selecting one. However, this process has failed to adequately serve the autistic population, and for this reason we seek to work around it. In addition to specifically tailor to the autistic population, our job-matching algorithm differs from existing ones in that it relaxes the restriction on the one-to-one matching between users and employers so that those who perform poorly in the existing system can still have unique jobs suggested to them. Instead of looking at the similarity between profiles in a single vector space, we consider the similarity in the *aptitude* and *interest* vector spaces separately and use this information to define a *stable matching*.

3 OUR JOB-MATCHING ALGORITHM

The central part of our job-matching algorithm is an extension of the Gale-Shapley stable-matching algorithm [6] so it may be applied in cases where the basic algorithm cannot be. While our algorithm is not the first extension of the Gale-Shapley algorithm, it is the first to use the algorithm to generate a *ranking* rather than a single match. While such a ranking is not useful or even meaningful to all applications of the Gale-Shapley algorithm, it works well with the assumptions made in this problem of matching adults with autism with potential employers. The idea of generating a *ranking* makes sense in this context, since it is based on the assumption that the information the model has is mostly accurate information, but may have incomplete information relating to the preference of the user making their choice. Users then complete the missing information by selecting their preferred employer from the ranking. We can assume that getting this information after the ranking is done does not violate the integrity of the results because the algorithm is a strategic proof from the perspective of the users [5]. This means it is in the best interest of the users to accurately give their interests as far as they can. Our algorithm could also potentially be applied to other problems where the objective is to approximate a stable solution in a two-sided market. Technology that could use such problems includes applications ranging from dating apps to tools analyzing various financial markets. This novel ranking idea may help fill in missing information that was missed during other parts of an automated process when tackling these problems.

3.1 Server and Client Specifications

Records containing user profiles and employer profiles, as well as additional information associated with them, are stored on the server. The record for a particular user contains the username, password, a text description detailing whatever the user is offering, a binary flag specifying if they are an employer or not, a sequence of double precision floats representing the user’s *profile vector*, and a flag specifying if the user has finished creating their account (see Figure 1 for an example). Similarly, an employer profile contains the corresponding information about an employer. It is on this server that our job-matching algorithm is performed. The server

Username Password

Description

Is an Employer ☐

Profile (Interest and Aptitude) Vector

Interests

Aptitude

Figure 1: A graphical representation of a system record

is designed so that a user (an employee, respectively) can communicate with it using a specifically designed client program, and it supports six different contexts through which the client can send messages. These contexts are *Home*, *Register*, *Login*, *Delete*, *Update*, and *Match* (see details in Section 3.1.1).

3.1.1 The Server's Contexts. Sending a message to **Home** is used to establish a secure connection with the server. (See Figure 2 for the layout of the client-server architecture of the proposed system.) Messages to all other contexts are encrypted as they, in the very least, include the user's password, and often contain other potentially sensitive information as well.

The **Register** command is used to create an account by sending to the server the username and password for the account that the user wishes to create. As long as the username is not associated with any existing records, a default record will be created containing that username and password. By default, the text description is an empty string, the user is not an employer, the vector is set to random values, and the account is not complete.

The **Login** command retrieves the record corresponding with the username that is sent to the server and the user's record is forwarded to the client if the password matches what is in the record. All the remaining commands will only be undertaken if the password matches what is in the record for the username that is given.

Delete causes the server to remove the record corresponding with the username that is passed to it, and **Update** replaces the record for the user with one corresponding to a serialized record sent with the request, and all values in the record other than username and password are set.

Finally, **Match** returns the usernames and descriptions of the user's top ranked matches. Before the user's Match request may be satisfied, their account must be marked as being completed, and only completed accounts will be considered when running the matching algorithm.

3.1.2 Overview of the Client. The client can be run from a JavaFx application and communicates with the server on behalf of the user. This application provides a user interface with buttons and text fields so the user may provide the client the information it needs to send its requests in a format that the user understands. The application also maintains a working model of the user's intent based on user input. To guide the user through creating and using their profile, the application is divided into several pages, i.e., XML documents, that are navigated through by pressing buttons in the user interface (see the configuration in Figure 3). Each page

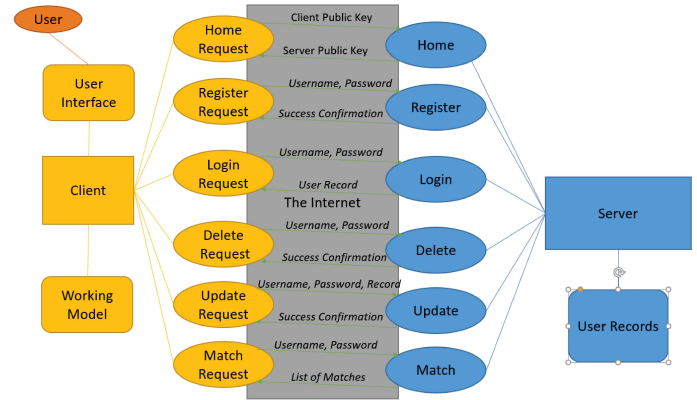


Figure 2: The user's computer (on the left) communicates with the server (on the right) over the Internet through the client. Messages (in italics) over the Internet are encrypted.

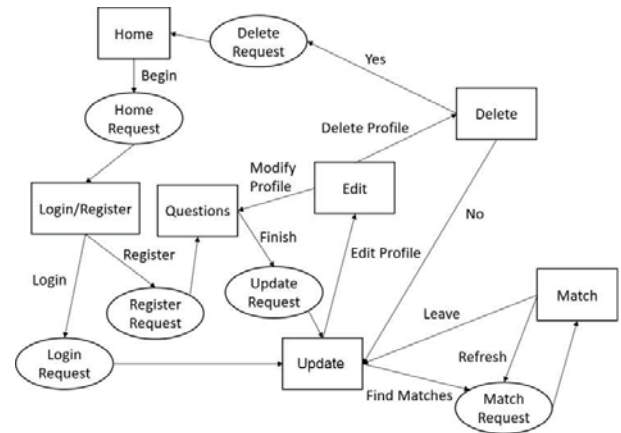


Figure 3: The structure of the client app. Rectangles are pages, i.e., XML documents, ovals are requests to the server (one for each context), and arrows are buttons.

defines what the user sees, including text, buttons, and text fields. Some button presses also cause the client to send a request to the server, in which case the application must wait for the server response before it changes pages.

3.2 Novel Extensions to the Gale-Shapley Algorithm

Our implementation of the Gale-Shapley algorithm, which is designed for matching students with universities for college admissions and matching couples for marriage, is augmented with two novel extensions. This includes (i) a routine for recursively applying the algorithm in order to generate a *ranking* for a user instead of just a single match, and (ii) a *filter* so that the algorithm can both run faster and run even when the number of users and employers is *not* equal. As there may be some inaccuracies in the stable matching due to an inability to perfectly capture all information about a user or employer that may be of interest to the opposite party, a *ranking* of matches is given rather than just a single match so that the user may choose for themselves along the matches. Our

method of augmenting the Gale-Shapley algorithm so that it may be used for ranking is unique to our work. This ranking method requires a *filtering* system, which is another augmentation to that algorithm that is also needed to ensure that the Gale-Shapley algorithm can be applied even with a *dynamic* set of users and employers. This relaxes another restriction on the Gale-Shapley algorithm, which requires a static set of users.

3.3 Stable Matchings

While this is not the first matching algorithm to be applied to helping adults with autism find employment [3, 16], it is the first where all the implementation details are publicly available, and the matching algorithm used is *novel*. It is based on the *Gale-Shapley algorithm*, but it is augmented with original features.

A matching algorithm can fulfill different criteria, with the Gale-Shapley algorithm finding the *single stable matching* which is optimal for one of the two parties that it is matching [6]. A matching is defined as a one-to-one mapping between two parties, with the pairing between a user and an employer called a *match* in our case. Furthermore, a matching is *stable* if no two participants may both have a *higher ranked match* with each other than who they were already paired with in the stable matching. (Figure 4 gives examples of an unstable and stable matching for the same data set.) The reason we are choosing to find this stable matching and filtering rather than fulfill a different criterion is because our matches are *non-binding*, with either party being free to accept or reject the match, since the user still needs to apply for the job afterwards and it's still up to the employer's discretion to accept the application. If a stable matching is accurate, then both the user and employer should have no reason not to accept the match, since they would not be able to find a better partner they were matched with and who would also reciprocate their choice. If the recommended employer for a user is not a pairing from a stable matching, it may be in the advantage of the user or employer to ignore their matching, defeating the purpose of suggesting that match.

For a given dataset, *multiple* stable matchings may exist, and it is possible to find the *optimal* stable matching according to arbitrary objectives [19], but we are choosing to just use the one found by the Gale-Shapley algorithm. The linear programming algorithm necessary to find other stable matchings is both harder to implement and slower than the Gale-Shapley algorithm, so there must be a compelling reason to optimize a different objective in order to justify using the more complex algorithm. We consider finding the optimal stable matching from the perspective of the user's to be a good objective for the sake of benefiting the autistic community and using the Gale-Shapley algorithm is sufficient to reach it.

3.4 Creating Profiles for Matching

Before users can be matched with employers, individuals in both parties need to create profiles for themselves within the system. When making a profile, someone first specifies if they are looking for or offering employment, which determines if they are a user or an employer, respectively. In either case, the user or employer will be walked through more questions to continue building their profile. Users will be asked questions to figure out both what jobs interest them and what skills they have. (See Appendix A for the

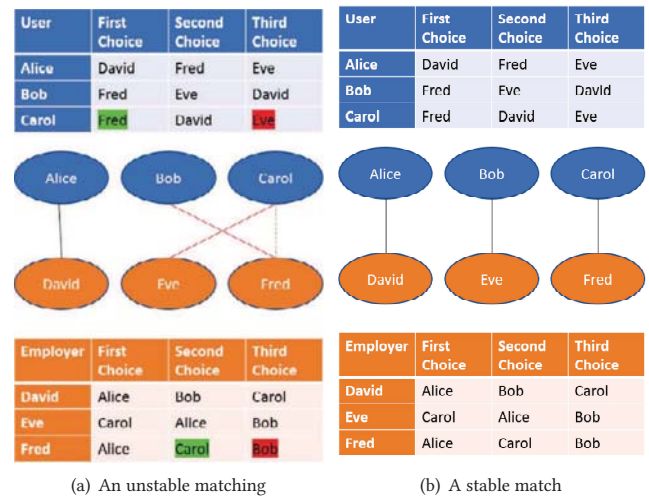


Figure 4: The red bold lines in 4(a) show an example of an unstable matching for a given set of rankings. It is unstable, since Carol and Fred would rather match with each other (dashed line) than their given match (Eve and Bob, respectively), whereas 4(b) is a stable match for the same data set.

sample set of questions for user and employers created for the job-matching system.) Based on their responses to the questions, a *numerical record* will be generated with different fields corresponding with different tasks. This numerical record has two parts, an *aptitude* portion corresponding with *skills* and an *interest* portion corresponding with interests. Employers, on the other hand, will be asked questions about the job they are offering to determine the qualities required to obtain and excel at the job, and what qualities the job has which may be of interest to a user. A *numeric record* is generated for them as well whose fields directly correspond with those in a user's record—for requirements, the larger the number in the employer's field means the more of the corresponding skill is demanded from the user. This defines the employer's *aptitude vector*. Similarly, the same number in the user's corresponding field means it matches a user's *interest* in that respect, defining the employer's interest vector. The system for guiding users through creating their profiles and storing the corresponding records is detailed in Section 3.1. When employers and users are matched with each other, users will evaluate employers with similar profiles, and vice versa.

3.4.1 Searching for Matches. After a user has created their profile, s(he) has the option to look for jobs by pressing a button. This will trigger the *matching* algorithm which will then return a list of ranked jobs to be displayed for the user. The list includes the *name* of the employer offering the *job* and information about the job. The user can always re-press the button to *refresh* the list of jobs in case activity from other users and employers has changed the results, which just runs the matching algorithm again for whatever user and employer profiles are currently in the system, but there is also a second button to *reject* all the jobs in the list. Choosing the latter will change the user's record to include flags that specify that the user is not interested in those jobs and will not include them in further matches, and also *update* the user's interest components of their profile to be further away from the rejected

profiles. The specific details in how this is done is described in Section 3.6. When running the matching algorithm, users will *rank* employers with similar *interest vectors* higher, whereas employers will rank users with similar *aptitude vectors* higher. The *interests vectors* (*aptitude vectors*, respectively) of the employers (users, respectively) measure how much the users (employers, respectively) are satisfied with the employers' requirements (users' qualifications, respectively) as used by our job-matching algorithm.

3.4.2 Filtering Profiles. Our *matching algorithm* is based around the Gale-Shapley algorithm, but includes some additional steps to ensure that prerequisites for using the Gale-Shapley algorithm are satisfied, and so that it can be used to generate a *ranking* rather than just a single match. First, *stable matchings* are only defined when the two parties being matched have the same number of participants, so the Gale-Shapley algorithm itself requires that there be the same number of user and employer profiles being matched. To ensure this, a *filtering* scheme that guarantees an equal number of users and employers is applied so only the user and employer profiles which are predicted to be most likely to impact who the target user is matched with are considered for matching. Specifically, the filtered employers will consist of jobs the target user is most likely to *apply* for based on being closest to what he is most *interested* in, and jobs the target user is most likely to *succeed* at based on their *aptitude* meeting the employer's *requirements*. (Table 1 shows a number of sample questions used for creating the *aptitude vectors*.) Meanwhile, the filtered users will be the users the target user is most likely to compete with when going after jobs they are interested in based on having similar *interests*. (Table 2 includes a number of sample questions aimed for finding the *interest* of a potential employer and user.) Our filtering scheme is based on both requesting the n (≥ 1) different employer profiles representing the jobs the target user is estimated to be the mostly likely to succeed at, and the m (≥ 1) employer profiles that the target user is predicated to be the most interested that are not already being considered with the previous request. Multiple studies have confirmed that with randomly generated rankings the expected ranking for the final match is $\log(\text{Number of Profiles})$ [18]. For this reason, we advise setting n and m to be around \log of what is the estimated maximum number of users. In our implementation, we set $n = 100$ and $m = 50$, arbitrary values from a range estimated to be high enough to obtain accurate results and small enough to run in a reasonable amount of time.

To calculate the n jobs the user is most likely to *succeed* at, we first calculate the discrepancy between a user and an employer as taking the sum of squared differences in all the fields (represented as components of a vector) in the employer's profile where the employer's requirement exceeds the user's skill as denoted by the corresponding field in their profile and as shown in Equation 1.

$$\text{Div}_{App}(U, E) = \sum \text{ReLU}(E_{A_i} - U_{A_i})^2 \quad (1)$$

where $\text{ReLU}(X) = X$, if $X \geq 0$, otherwise, $\text{ReLU}(X) = 0$, and E_{A_i} is the i^{th} component of the Employer's *aptitude vector*, and U_{A_i} is the i^{th} component of the User's *aptitude vector*.

EXAMPLE 1. Assume that under a particular scheme, a user vector with *aptitude* components is $\langle 1.2, 2.3, 4.9 \rangle$, which represents their *interview skills*, *noise tolerance*, and *loyalty*, while an employer

with *aptitude* vector is $\langle 3, 2.5, 2 \rangle$, meaning the employer puts significant emphasis on *presentation during interviews*, the environment is *moderately noisy*, and (s)he weakly desire *company loyalty*. The user's fit for the job would be interpreted that (s)he is *sufficiently loyal*, and would only need *minor accommodations* at most to handle the noise in the environment, but (s)he would need to work on his/her *interview skills* significantly so that (s)he likely succeeds at applying to the job. This corresponds with component-wise discrepancy of 3.24, 0.04, and 0, which sums to 3.28 for the *total discrepancy*. \square

As shown in Example 1, the discrepancy represents extra labor the user must expend to reach the demands of the job, or additional accommodations the employer must make to be accessible to the user, since a higher discrepancy means a user is a worse fit for the job. We can then take those n employers with *minimal* discrepancy between them and the target user by ranking the profiles in order of *increasing* discrepancy and keeping only the top n in the ranking. In the case of a tie where two employers have the same discrepancy from the target user, the first of the two employer profiles to be created is given priority in the ranking, assisting employers who have been waiting longer to be reached in the system. The top m profiles that the target user is predicted to be the most *interested* in are calculated in a similar way, but with a different formula for discrepancy. This is done by comparing the fields (again represented as components of a vector) related to *interest* instead of those relating to *skill*, and including all fields in the sum, not just those where the employer's value is greater than the user's as shown in Equation 2.

$$\text{Div}_{Int}(U, E) = \sum \text{ReLU}(E_{I_i} - U_{I_i})^2 \quad (2)$$

where E_{I_i} is the i^{th} component of the Employer's *interest vector*, and U_{I_i} is the i^{th} component of the User's *interest vector*.

Equation 2 is equivalent to the Euclidean distance between the *interest* fields of the profiles as modeled as vectors in a normed space. This *discrepancy* represents the divergence between a user's ideal job and the given job. A potential scheme for *interest* includes consistency of tasks, and the social culture of the workplace, with higher values denoting more consistency in tasks and a more prominent social culture.

EXAMPLE 2. Assume that the *interest* vector of an adult with autism is $\langle 2.4, 0.9 \rangle$ under our encoding scheme, meaning (s)he would like *strong consistency* in their tasks and would prefer not to *interact with others* while working. It is further assumed that the *interest vector* of an employer is $\langle 3.1, 2.2 \rangle$, suggesting that the work is quite *repetitive* and that there is a *moderate social culture* in the workplace. The component-wise discrepancy is 0.49 and 1.69, which sums to 2.18. \square

If there are less than $n+m$ ($= 150$ in our implementation) employer profiles in the system, then all of them will be considered, and these calculations for *discrepancy* to obtain the top n and m employer profiles can be skipped. As a result, either $n+m$ employer profiles will be considered after filtering is applied, or no filtering will be applied and all of them will be considered, in which case we define E as the total number of employer profiles.

$$N = \min(n + m, E) \quad (3)$$

Table 1: Sample questions used for creating the *aptitude* vectors

User's Feedback			Objectives	Employer's Feedback		
Question	Response	Score		Question	Response	Score
How are your interview skills?	Poor	1.2	Interview Skills	How important is presentation during interview?	Very	3.0
How well can you function in a noisy environment?	Manageably	2.3	Noise Tolerance	How noisy is your workplace?	A little bit	2.5
How loyal are you?	Extremely	4.9	Loyalty	How important is company loyalty?	Slightly	2.0

Table 2: Sample questions used for creating the *interest* vectors

User's Feedback			Objectives	Employer's Feedback		
Question	Response	Score		Question	Response	Score
How much do you like task consistency?	I like strong consistency	2.4	Task Consistency	How consistent are the job tasks?	They are quite repetitive	3.1
What social culture do you desire in a workplace?	I prefer not to interact w/ others while working	0.9	Social Culture	What is the social culture in your job like?	There is a moderate social culture	2.2

where N is the number of filtered employer profiles and is also the number of filtered user profiles to be considered so that we can ensure that the number of users being considered is equal to the number of employers.

Next the $N-1$ user profiles with the most similar *interests* to the target user are considered so that, together with the target user, N user profiles will be considered, ensuring that the same number of user and employer profiles are considered after filtering. An example of the result of filtering with $n = 2$ and $m = 1$ is shown in Figure 5.

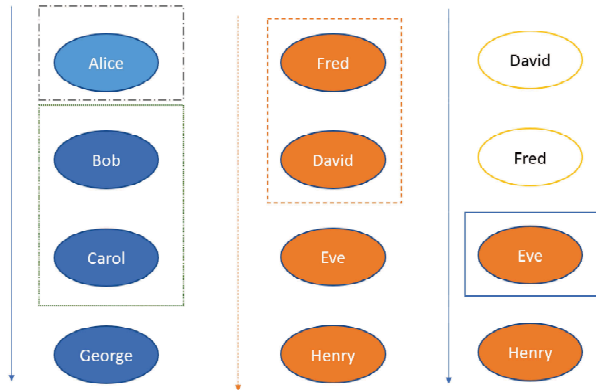


Figure 5: Users (blue) and employers (orange) being considered for matching are enclosed in rectangles. Each column is a sorted list, with the arrow on the left showing if the list is sorted by *interest discrepancy* (blue/solid) or *aptitude discrepancy* (orange/dashed). It is assumed that Alice is the target user, so all discrepancy is measured relative to her. Note that $n = 2$ in this case, so Fred and David are enclosed in the orange (dashed) rectangle, and $m = 1$, so Eve is enclosed in the blue (solid) rectangle after David and Fred are passed over due to already being considered. Alice herself is the grey (dot and dashed) rectangle, and $n + m - 1$ users closest to Alice are in the green (dotted) rectangle. Together, three users and three employers are being considered, so a matching is defined.

The reason users with a similar *interest* are considered is because they are the most likely to compete with the target user for their preferred job and thus affect the results of the Gale-Shapley algorithm. The corresponding *discrepancy* in *interest* is calculated in the same way as the m employers the user is most likely to be interested in, by calculating the Euclidean distance between their interest records, and keeping those with the lowest scores. To ensure $N-1$ users can always be filtered, $n+m-1$ mock user profiles are included within the system in addition to real user profiles. These mock profiles do not correspond with any individuals and just exist to ensure the algorithm can be applied. They are randomly generated, but form a distribution that matches that of real user profiles, including potential profiles that would correspond with non-autistic individuals in order to simulate wider competition. While these mock profiles may influence the results of the algorithm as they simulate competition, they will never be target users, and will never compete with actual users when users apply for jobs after being matched. The fact that the filtering never returns more than the requested number of profiles means that the matching that follows will run in *constant time* relative to number of profiles, improving over the *quadratic time* of the unconstrained Gale-Shapley algorithm, though the computational time for filtering grows *linearly* with the number profiles. As a result, the overall time complexity for matching a single user is *linear*.

3.5 The Modified Gale-Shapley Algorithm

The next pre-requisite the modified *Gale-Shapley algorithm* needs before it can run is to be provided information about how each user being considered will rank each employer profile being considered, and vice versa. For this, users rank employers by how *interested* they are in them, whereas employers rank users by how likely they are estimated to *succeed*. These ranks are calculated in the same way they were calculated during the *filtering* process, by sorting the calculated discrepancy so that those with the *lowest discrepancy* are most preferred. With the rankings generated, the modified Gale-Shapley algorithm is now applied to find a *stable matching*.

When the algorithm starts, users are labeled as being not considered matched, but all users are labeled as being considered matched

when it stops. The modified Gale-Shapley algorithm consists of applying the following loop, called the **Gale-Shapley Loop**, that matches and un-matches users until every user is considered to be matched with an employer. At that point these matches are now considered as the official matches which are returned.

1. Every user who is not considered to be matched is considered as a potential match to their current top ranked employer.
2. Each employer who becomes matched to their top ranked user is being considered as a potential match. The other users who were being considered to them will no longer be considered to be matched, and will now consider their next top ranked employer as their current top ranked employer.

EXAMPLE 3. Figure 6 shows a complete run of the modified Gale-Shapley algorithm on a simple dataset. Step 1 is the initiation, and every following step alternates between Step 1 and Step 2 in the Gale-Shapley loop. *Potential matches* are blue (asterisk/dotted), *matches* are green (plain/solid), and *rejected matches* are red(xed/dashed). This example does not show any cases of former matches becoming rejected (going from green to red), but such behavior is possible. □

3.6 Generating Ranked Results

The employer the target user is matched with is returned as his top suggested employer. To generate the rest of the ranking, assume that the user was not interested in the most recent employer that was suggested to them. To take into account this scenario, the components in the vector representation that measure the user's interest would be updated to be further away from the components in that employer's record. This is done by subtracting a weight¹ multiple of the component of an employer's profile vector from the corresponding component in the user's profile vector for each component representing interest as shown in Equation 4.

$$U'_I = U_I - w \times (E_I - U_I) \quad (4)$$

where U_I is the target user's *interest* vector, E_I is the matched employer's *interest* vector, w is the weight given to negative feedback, and U'_I is the target user's updated *interest* vector.

EXAMPLE 4. Consider the *interest* vector of an adult with autism as shown in Example 2, which is $\langle 2.4, 0.9 \rangle$, and the *interest* vector of the matched employer, which is $\langle 3.1, 2.2 \rangle$. Further assume that the weight w is 0.5. Hence, the difference between the interest vectors of the two profiles is $\langle 0.7, 1.3 \rangle$, and the updated user interest vector in his profile is $\langle 2.4, 0.9 \rangle - 0.5 \times \langle 0.7, 1.3 \rangle = \langle 2.05, 0.25 \rangle$. A visual representation of this change is shown in Figure 7(a). □

The matched employer will also not be further considered when filtering employers. With this in mind, the rest of the matching algorithm can be re-applied with the updated information, i.e., updated interest fields and ignoring the last recommended profile, in which case it will generate a new matching and return a new match. (See Figure 7(b) for a new set of candidates to be considered for matching.) This new match can be returned as the next suggested

¹The value of the weight is determined empirically with the goal of having users choose higher-ranked matches. This can be determined by finding an approximately optimal solution to minimizing the aggregated rank users' choices based on experimental data.

employer. The process can in theory be repeated until every employer has been ranked, but it only needs to be applied until the specified number of employers to be displayed to the user have been ranked, at which point they are displayed to the user. This same process is used to update a user's profile if the user rejects all the matches, making it so that if k results are listed at a time, then rejecting the results would cause ranked results $k+1$ through $2k$ according to the original profile vector to be displayed instead, pressing it again would display results $2k+1$ through $3k$, and so on. This process of iteratively applying the modified Gale-Shapley algorithm on filtered results to create a ranking is *novel*. The optimal value of k depends on what is practical to display on the screen to the user while still retaining ease of use, we have chosen $k = 5$. Shown below is the modified *Gale-Shapley* algorithm.

Algorithm. Modified Gale-Shapley Matching Algorithm (User, Employers, Users)

Begin

1. Let *Target User* be *User*, *Potential Employers* be *Employers*, and *Potential Users* as *Users* excluding the *Target User*
2. For ($i = 0$; $i \leq k$; $i++$) DO
 - (a) Apply the filter on *Potential Employers*, *Potential Users*, *Target User* to obtain N *filtered-employers* and N *filtered-users*.
 - (b) Call *Gale-Shapley algorithm* (*Filtered Employers*, *Filtered Users*) [5] to obtain the *Stable-Matching map*.
 - (c) Select the match for the *Target User* from the *Stable-Matching map* and return the *Matched Employer*.
 - (d) Let the i^{th} ranked match be the *Matched Employer*.
 - (e) Update the interest vector of *Target User* as $TargetUser_I - w \times (Match_I - TargetUser_I)$, where X_I denotes the interest vector of the user or employer, and w is the weight.
 - (f) Let *Potential Employers* be the *Potential Employers* without the *Matched Employer*.
3. Return the ranked matches

End

4 SYSTEM SIMULATION AND RESULTS

While we are supposed to test our solution on end users to see if the designed system is superior to other solutions in practice, we prove that some aspects of our solution are superior to other solutions instead, at least under certain conditions. Specifically, we have tested our matching algorithm in a simulated environment, showing that our matching algorithm is superior to comparable algorithms for the case that was simulated. This simulation was coded in Java and ran in the IntelliJ coding environment.

4.1 The Simulation

For the simulation, we created a workable system where users and employers continuously enter the system and leave when they are either hired or hire someone, respectively. Users in the system request matches and apply to one of the matches suggested to them. After a user applies to the job that an employer is advertising, the employer will decide whether to hold, reject an applicant, or eventually hire an applicant that they held. If a user is hired, they will be removed, along with the employer, and the count of successful hires will be incremented. The simulation ceases after a set number

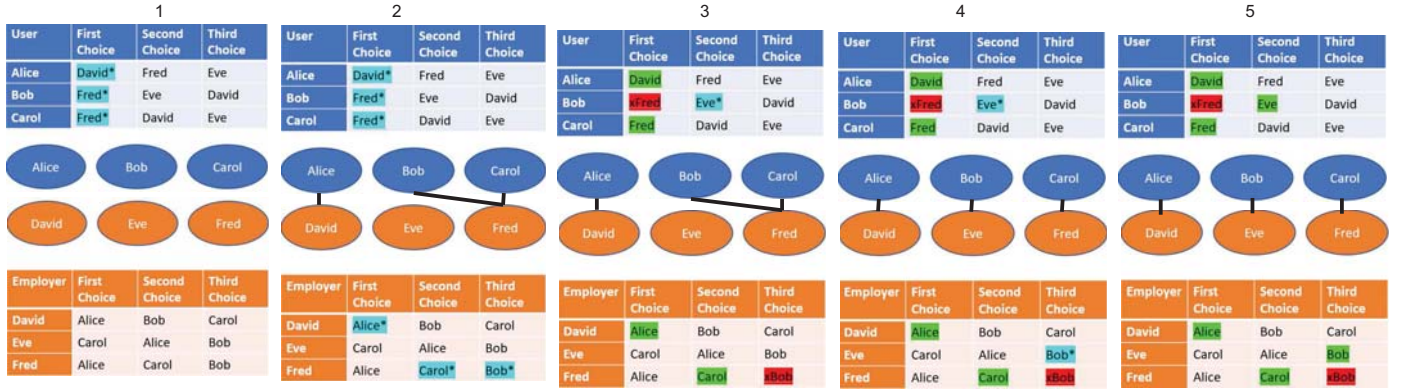


Figure 6: An example of running the modified Gale-Shapley algorithm

of steps, which was 100 in our simulation, and the number of successful matches is returned. This count is used to compare different matching schemes, which are treated as individual algorithms in this simulation.

During the simulation process, we considered four different matching schemes, denoted “Matched”, “Interest”, “Aptitude”, and “Mixed”. **Matched** refers to ranking users using our own matching algorithm, whereas **Interest** and **Aptitude** rank users by interest and aptitude divergence, respectively, and **Mixed** refers to ranking users by the sum of aptitude and interest divergence when they are being recommended to a target user. As these different schemes are based on making matches from the same distribution of vector spaces, they are comparable, and we know that differences between these results must be due to the matching algorithm itself rather than due to what data they utilized. Our matching scheme differs both from the comparable schemes and other existing matching schemes as mentioned in Section 2 in that the former considers the competition along the *two* aspects to find stable matchings, whereas the latter just try to optimize the discrepancy between employer and employee. For each matching scheme, we ran the simulation 100 times and calculated the *average* number of successful matches, as well as the variance of successful matches in the sample, so that *null hypothesis* testing could be performed. Based on these statistics, the null hypothesis that our matching scheme performs equally well or worse to comparable matching algorithms in terms of average successful matches under the parameters of the simulation is *rejected*, proving that our method is *superior* in the case that was simulated.

Every time the *step event* is executed, new users or employers will be generated from the same *geometric distribution* to simulate users continuously entering the system. This geometric distribution is defined by its *stopping probability*, which is 0.3 in our simulation². Each time after a user or an employer is generated, an event for making their profiles will be queued.

Once a user has created a profile, the user will queue an event to request matches. Out of the list of given matches, the user applies for the one that (s)he likes the most in terms of interest as based on the true vector’s values. The user will then wait until (s)he has

received the rejected or hired confirmation. If rejected, the user will request matches again by queuing another match event and the application process will repeat. A constraint is that a user will not apply to the same job twice. If a user has applied to every job, (s)he will wait before refreshing the job list by queuing another match event to be executed during the next step.

In addition to the *aptitude* vector, an employer will have a *threshold* for each component of the vector, and these thresholds denote the requirements for a job. After an employer creates his/her profile, the employer will queue a *wait event* and stick around until (s)he has users in their application queue. If an employer has applications in the queue, (s)he will test the first user in the queue according to the thresholds and remove him/her from the application queue. If a user is tested and the user does not meet the threshold in some component, meaning the threshold for that component is greater than the value in the user’s true *aptitude vector*, the user will be rejected. If the employer rejects a user while not holding any users, then the employer will have to lower the threshold to be between the previous value and the value of the rejected user’s component according to a *linear function*, simulating the employer becoming more willing to accommodate as they are unable to find someone who can fulfill the requirements. In our simulation, the *adjustment weight* was set at 0.5, meaning that new threshold is set halfway between the old threshold and the user’s component. If the tested user meets all the thresholds, (s)he will be held if no user is currently being held.

Once an employer has held an applicant, (s)he will wait a fixed number of steps before hiring the applicant. In our simulation, the wait lasts ten steps. If a tested user both has a lower discrepancy than the held candidate and is accepted based on the thresholds, then (s)he will be held and the previously held user will be rejected. However, if a tested user has a lower discrepancy than the held candidate but is rejected due to the fact that (s)he is unable to meet the threshold in some component, the employer will lower the threshold, simulating the employer moving to accommodate users that are generally better for the position, but are not currently being accommodated. Once a user is hired, both the user and the employer will be removed from the system and will not be considered by the matching algorithm.

²0.3 is the probability that generation will stop after each user or employer is generated and the step event ends.

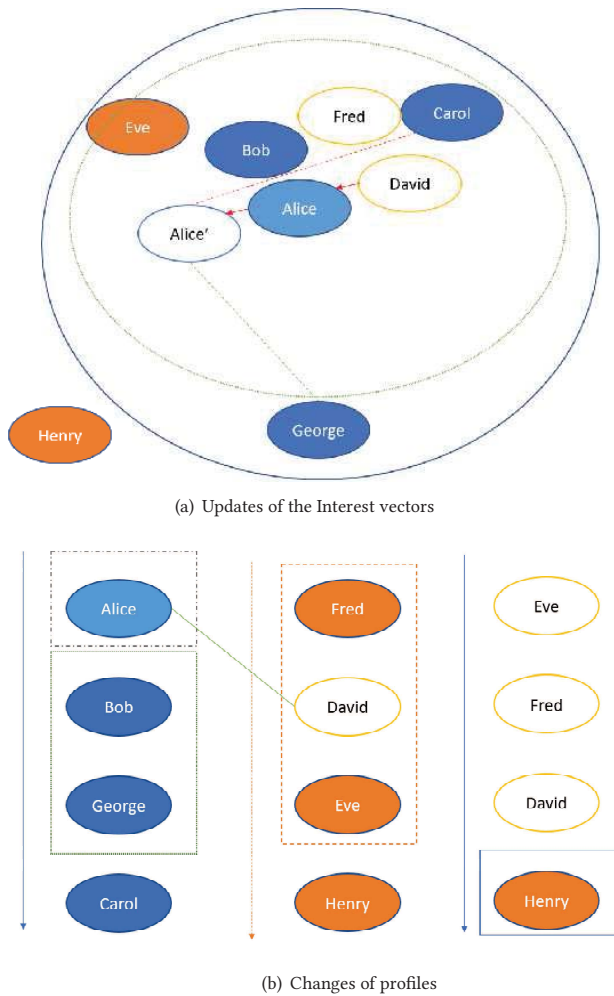


Figure 7: (a) The green (dotted) circle encloses the closest users in terms of interest to Alice according to her original interest vector, while the blue (solid) circle encloses the closest employers. Alice' interest vector is at the position of the new vector after moving away from David, her top-ranked match. Alice's interest vector is closer to George than Carol, changing who is considered in the next match. However, Eve remains closer to Alice than to Henry. (b) The filtered results for the second match. Note that the *interests* sorting has changed from the first match, but not the *aptitude* sorting. Since David was already found as a match, he is no longer being considered, changing who is considered in both the n closest by *aptitude* and the m closest by *interest*.

4.2 The Simulation Results

The results obtained from our simulation run are shown in Table 3. Psychology uses a p -value of 0.05, meaning that if the probability of the z -score under null hypothesis is less than 0.05, then the results are *statistically significant*, and the null hypothesis should be rejected. In this case, all the probabilities are well below 0.05, and thus we can safely claim that our method is more effective than comparable methods for the given parameters.

Table 3: Results obtained from the simulation

Scheme	Matched	Interest	Aptitude	Mixed
Average	69.72	48.99	42.10	48.47
Variance	130.57	82.68	84.20	96.82
Z-score for Matched \geq Null Hypothesis	N/A	4.49	5.96	4.46
Probability of z -score $<$ Null Hypothesis	N/A	3.62×10^{-6}	2.0×10^{-9}	4.22×10^{-6}

4.3 Observation

While the simulation was only run for the given parameters, these parameters were chosen arbitrarily within a simulation scheme designed to emulate the behavior of real-world agents. Since we lack real-world data to fill in the parameters, and it is not possible to run the simulation on every possible combination of parameters, we can assume that the results from this set of parameters are as valid as any other arbitrary set of parameters. This is enough to establish the potential superiority our method may have it applied in the real-world.

5 CONCLUSIONS

Anecdotal reports from companies with programs that bring in workers with autism show that having such workers changes the attitudes of their co-workers [17]. It opened the minds of these workers to not only being accepting of a more diverse population, but to also consider different problem-solving strategies. The fact that employees with autism are now financially independent also decreases financial strain on relatives and their generally increased well-being reflects positively on everyone they interact with. In this paper, we have proposed a job-matching algorithm for adults with autism and potential employers that connects adults with autism with the necessary job skill required by the potential employers and contribute to the autistic community and companies who can benefit from hiring diverse work force.

Since our job-matching algorithm is agnostic to whether or not a user actually has a diagnosis of autism, it can also be used to assist individuals who do not have a diagnosis but face similar obstacles to finding employment. Some other disorders were potential employees with these disorders may have similar concerns include sensory processing disorder, social (pragmatic) communication disorder, language disorder, obsessive compulsive personality disorder, attention deficit disorder, and social anxiety disorder. While our proposed job-matching system is designed with autism in mind, it could be expanded to include questions relating to even more disorders. As the system is designed to be competitive, anyone could potentially sign up for it, so it could potentially serve as an alternative means of finding employment for adults with other disability in general.

A THE APTITUDE AND INTEREST SCHEMA

The *Interest* component consists of *Social Culture*, *Repetition*, *Salary*, *Location* and *AQ* score, whereas the *Aptitude* contains *Light Sensitivity*, *Sound Sensitivity*, *Interview Skills*, *Loyalty*, *Trustworthiness*, *Attention to Detail*, *Office Skills-Social/Technical*, *Fine Motor Skills*, and *Gross Motor Skills*. Table 4 depicts some of the sample quizzes.

Table 4: Sample queries of the Aptitude and Interest schema

Question	Em- ployee	First Answer	Second Answer	Third Answer	Min Value	Lower Value	Upper Value	Max Value
How much do you like task consistency	Yes	I can't stand repetitive tasks.	I like some sense of routine, but with some variety as well.	I prefer to just perform a single task.	0	3	6	9
How consistent are the job tasks?	No	Everyday is different at our company.	While the bulk of the work is similar daily, new problems are always arising the need to be solved.	Our work is extremely repetitive.	2	4	8	10
How are your interview skills?	Yes	I perform poorly during interviews.	I am not skilled at interviews, but do not consider the interview process to be a major obstacle.	I am highly skilled at demonstrating my soft skills during interviews.	0	1	5	10
How important is presentation during interview?	No	For us, interviews are merely a formality and we do not evaluate a candidate based on their interview.	Interviews are useful for accessing the fitness of an employee, but are strictly secondary to resumes, portfolios, and referrals when assessing a candidate.	We closely monitor the body language of candidates and other features of the presentation during interviews in order to access soft skills.	0	2	7	10
What social culture do you desire in a workplace?	Yes	I prefer to work alone.	I am fine with working alone but enjoy the company of others.	I thrive off interacting with my coworkers.	0	2	7	10
What is the social culture in your job like?	No	People mostly keep to themselves.	Coworkers tend to maintain a casual relationship with each other.	Workers are expected to form intimate bonds with one another through their work.	1	2	8	10
How well can you function in a noisy environment?	Yes	I cannot function in a noisy environment.	Noise can distract or bother me, but I can manage.	I can effectively follow multiple conversations in a noisy environment.	-2	2	5	12
How noisy is your workplace?	No	The workplace is quiet, and workers can wear headphones if they please.	There is frequently ambient conversation and workers are expected to remain alert, but there is nothing out of the ordinary.	The workplace is filled with loud noises at all times	0	1	7	12

REFERENCES

- [1] R. Burke, M. Andersen, S. Bowen, M. Howard, and K. Allen. Evaluation of Two Instruction Methods to Increase Employment Options for Young Adults with Autism Spectrum Disorders. *Developmental Disabilities*, 31(6):1223–1233, 2010.
- [2] C. Cing. Development of Job Matching Algorithm with Collective Learning. Master of Computer Science, Universiti Tunku Abdul Rahman, August 2013.
- [3] Daivergent. <https://daivergent.com/data-services/recruitment>.
- [4] J. Dreaver, C. Thompson, S. Girdler, M. Adolfsson, M. Black, and M. Falkmer. Success Factors Enabling Employment for Adults on the Autism Spectrum from Employers' Perspective. *Autism & Developmental Disorder*, 50:1657–1667, 2020.
- [5] E. Dubins and D. Freedman. Machiavelli and the Gale-Shapley Algorithm. *The American Mathematical Monthly*, 88(7):485–494, August–September 1981.
- [6] D. Gale and L. Shapley. College Admissions and the Stability of Marriage. *The American Mathematical Monthly*, 69(1):9–15, January 1962.
- [7] C. Grob, D. Lerman, C. Langlais, and N. Villante. Assessing and Teaching Job-related Social Skills to Adults with Autism Spectrum Disorder. *Journal of Applied Behavior Analysis*, 52:150–172, 2019.
- [8] S. Guo, F. Alamudun, and T. Hammond. Résumatcher: A Personalized Résumé-job Matching System. *Expert Systems With Applications*, 60:169–182, 2016.
- [9] D. Hendricks. Employment and Adults with Autism Spectrum Disorders: Challenges and Strategies for Success. *Vocational Rehabilitation*, 32:125–134, 2010.
- [10] A. Jacob, M. Scott, M. Falkmer, and T. Falkmer. The Costs and Benefits of Employing an Adult with Autism Spectrum Disorder: A Systematic Review. *PLOS ONE*, pages 1–15, October 2015.
- [11] M. Koh and Y. Chew. Intelligent Job Matching with Self-learning Recommendation Engine. *Procedia Manufacturing*, 3:1959–1965, 2015.
- [12] J. Malinowski, T. Keim, O. Wendt, and T. Weitzel. Matching People and Jobs: A Bilateral Recommendation Approach. In *Proceedings of the 39th Annual Hawaii International Conference on System Sciences (HICSS'06)*, pages 137c–137c, 2006.
- [13] M. Parsons, D. Reid, J. Reynolds, and M. Bumgarner. Effects of Chosen Versus Assigned Jobs on the Work Performance of Persons with Severe Handicaps. *Journal of Applied Behavior Analysis*, 23(2):253–258, Summer 1990.
- [14] R. Rafter, K. Bradley, and B. Smyth. Personalized Retrieval for Online Recruitment Services. In *Proceedings of the BCS/IRSG 22nd Annual Colloquium on Information Retrieval (IRSG 2000)*, pages 1–13, March 2000.
- [15] A. Roux, J. Rast, K. Anderson, and P. Shattuck. National Autism Indicators Report: Developmental Disability Services and Outcomes in Adulthood. Indepth Report, Drexel Univ., May 2017.
- [16] Samhall. Samhall 2014 Annual and Sustainability Report. https://samhall.se/wp-content/uploads/2020/04/Samhall_anual-report2014.pdf, 2014.
- [17] E. Shein. Hiring from the Autism Spectrum. *CACM*, 63(6):17–19, June 2020.
- [18] G. Shi, Y. Kong, B. Chen, G. Yuan, and R. Wu. Instability in Stable Marriage Problem: Matching Unequally Numbered Men and Women. *Complexity*, page 5 pages, 2018.
- [19] J. Vande Vate. Linear Programming Brings Marital Bliss. *Operations Research Letters*, 8:147–153, June 1989.

On Efficiently Equi-Joining Graphs

Giacomo Bergami

Free University of Bozen-Bolzano

Bozen, Italy

gibergami@unibz.it

ABSTRACT

Despite the growing popularity of techniques related to graph summarization, a general operator for joining graphs on both the vertices and the edges is still missing. Current languages such as Cypher and SPARQL express binary joins through the non-scalable and inefficient composition of multiple traversal and graph creation operations. In this paper, we propose an efficient equi-join algorithm that is able to perform vertex and path joins over a secondary memory indexed graph, also the resulting graph is serialised in secondary memory. The results show that the implementation of the proposed model outperforms solutions based on graphs, such as Neo4J and Virtuoso, and the relational model, such as PostgreSQL. Moreover, we propose two ways how edges can be combined, namely the conjunctive and disjunctive semantics. Preliminary experiments on the graph conjunctive join are also carried out with incremental updates, thus suggesting that our solution outperforms materialized views over PostgreSQL.

CCS CONCEPTS

• **Theory of computation** → **Graph algorithms analysis**; • **Mathematics of computing** → **Graph algorithms**; *Paths and connectivity problems*; • **Information systems** → **Graph-based database models**; **Query languages**;

KEYWORDS

Graph Joins, Property Graphs, RDF Graphs, Near-Data Processing

ACM Reference Format:

Giacomo Bergami. 2021. On Efficiently Equi-Joining Graphs. In *25th International Database Engineering & Applications Symposium (IDEAS 2021)*, July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3472163.3472269>

1 INTRODUCTION

The larger availability of data, mainly as knowledge bases for scientific purposes [24], allows the combination of several different types of graphs. Transport networks [17], social network relationships [29], protein-to-protein association networks [26], and citation networks [27] are all examples of graphs. In the relational model, the join operator is the basic operation for combining different tables.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Association for Computing Machinery.

ACM ISBN 978-1-4503-8991-4/21/07...\$15.00

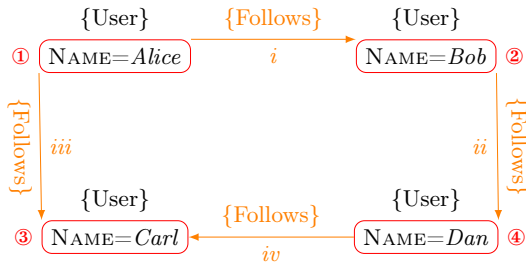
<https://doi.org/10.1145/3472163.3472269>

We would expect to use a similar operator for combining graph data. As evidenced by mathematical literature [13, 14], the desired graph products (see §6) must combine both vertices and edges of our (graph) operands and produce a graph as an output. Nonetheless, current literature meets no unanimous consensus for what is a graph join and, as illustrated in our running example, provides two distinct classes of graph operators, vertex joins and path joins. We now consider two distinct graph operands: a *Researcher Graph* (Figure 1a), and a *Citation Graph* (Figure 1b).

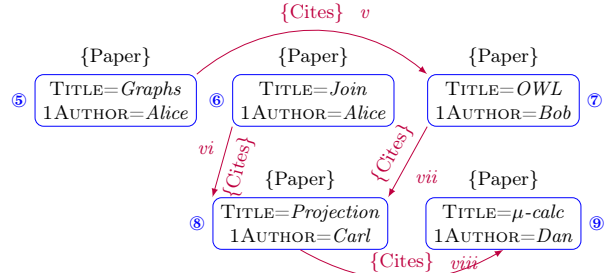
Authors in [9] propose a **vertex join** with link creation (Figure 1c): each social network user is matched to the paper where they appear as a first author. The black dashed edges in the figure provide the output of the link creation. In vertex join as graph fusion [15] (generalized in [4]), the two graphs should be distinct connected components of the same graph and the matching vertices are now fused into one single vertex (Figure 1d). **Path joins** such as SMJoin [10] are used for traversing one single graph operand at a time [11, 20], where each traversed or matched path $u \rightsquigarrow v$ is replaced by one single edge $u \rightarrow v$: this is possible because both the vertex set and edge set are considered as relational tables, and so the path traversal or match can be implemented as a multi-way join producing one single table, thus providing the new edges. The answer to “return the graph of papers where a paper cites another one iff. the first author 1AUTHOR of the first paper follows the 1AUTHOR of the second” requires a preliminary **vertex join with vertex fusion** and then a path join. For this reason, Figure 1e extends the output from the vertex fusion and creates additional links between the vertices having both a CITES and a FOLLOWS outgoing edge. This last approach is always possible as we might force any graph query language to generate a graph as an output (see §6).

At the time of the writing, both graph and relational query languages are the only way to combine vertex with edge joins for returning one single graph. Such languages, however, require the composition of multiple clauses and operators (§6), thus resulting in a non-scalable and relatively inefficient query plan for big data scenarios (§7). Furthermore, the lack of an explicit graph join operator for both query languages requires the rewriting of the graph join query each time the underneath graph schema changes [7], thus making the graph querying uneasy for real-world data integration scenarios such as the (semi-)automated ones described in [21]. We propose a novel graph equi-join algorithm over a secondary memory graph representation merging the vertex and path equi-join into one single step while traversing the graph. In compliance with the mathematical literature, the vertex join combines two distinct operand vertices into one and produces one single edge out of the graph traversal of two distinct paths (i.e., outgoing edges).

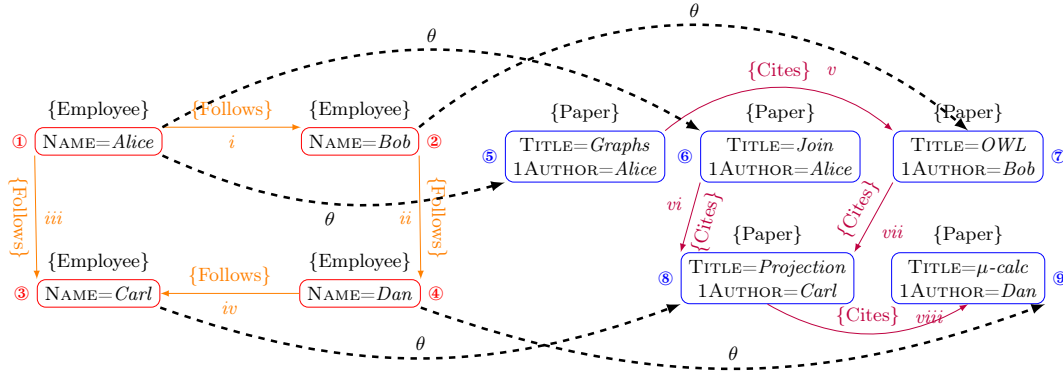
We now provide some use cases requiring a data combination from different sources for application and research purposes. For



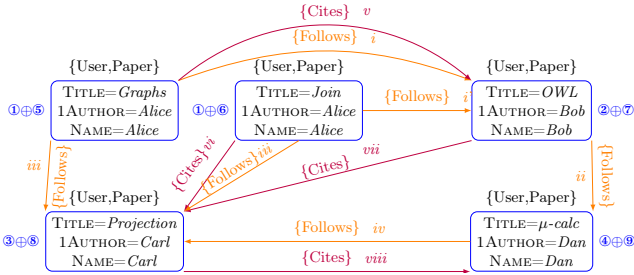
(a) Researcher Graph, Follower relations.



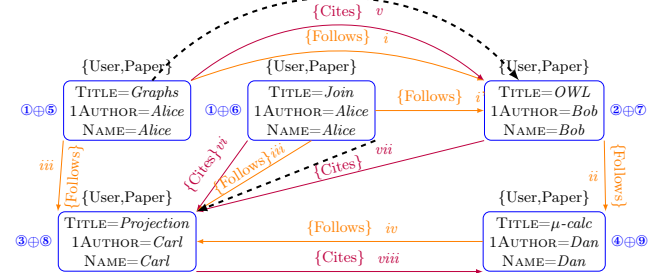
(b) Citation Graph, citation relations. Each paper has a first author.



(c) Vertex join with link discovery. The black dashed edges provide the output of the link creation.



(d) Vertex join with vertex fusion. The vertices that were previously linked are now fused into one single vertex.



(e) Path join over the vertices that have been previously fused via (d) with link creation (black dashed edges).

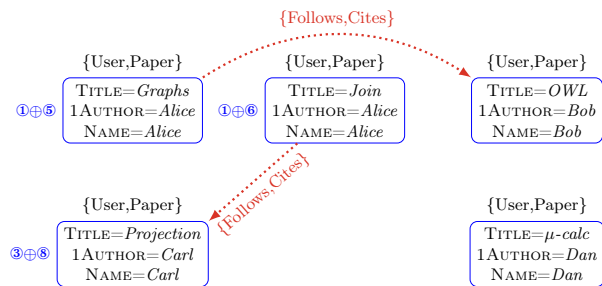
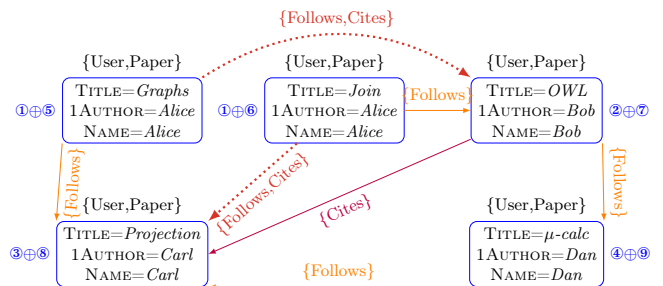
(f) Query with conjunctive semantics, $es = \wedge$: $\text{Researcher} \triangleright_{\text{Name}=1\text{Author}}^{\wedge} \text{Citation}$ (g) Query with disjunctive semantics, $es = \vee$: $\text{Researcher} \triangleright_{\text{Name}=1\text{Author}}^{\vee} \text{Citation}$

Figure 1: Example of a Graph Database containing two distinct graph operands, (a) and (b). Vertices are enumerated with circled arabic numbers, while edges are enumerated with roman numbers in italics. Dotted edges remark edges shared between the two different joins.

instance, the graph equi-join algorithm can be used for investigating multimodal demand in transportation network analysis [5]. We might be interested in joining together the networks of different transportation services, where nodes and edges represent the stops and path, respectively, to plan a long journey or balance the users’ demand. Edges from different operands might be combined in several different ways that, in this paper, we denote as **es** semantics.

Temporal networks are an appealing application for graph joins by (unique) node id: edges within a “snapshot” graph G_t represent interactions at a given time t . Nodes might represent users within social networks or computing agents, and interactions may represent text message or diseases’ spread. We may be interested in either selecting all the interactions occurring at every given timestamp or merging such interactions within one single graph. The first scenario will use the conjunctive semantics and the second one the disjunctive semantics.

The disjunctive semantics could be also applied to the bibliographic data scenario for answering the query “*For each paper reveal both the direct and the indirect dependencies (either there is a direct paper citation or one of the authors follows the other one in the Researcher Graph)*”. The resulting graph (Figure 1g) has the same vertex set as Figure 1f exploiting a conjunctive edge semantics, but they differ on the final edges.

Concerning our previous work [3], the main contributions are the following: we upgraded the definition of the logical model and simplified the definition of the join operators in [2], as well as upgraded the original GCEA algorithm to also support the disjunctive operation. For this novel algorithm, we preserve the original computational complexity optimality for the conjunctive case; we also discuss the computational complexity of the disjunctive semantics [2]. We expanded the experiments section, by both considering bigger graphs (up to 10^8 nodes) and more connected datasets (Kronecker Graph). Furthermore, we also showed how the basic procedures for joining graphs could be easily reused and minimally extended to support incremental graph join updates for the conjunctive semantics.

For our graph join operator, we specifically designed¹ two secondary memory physical models, one for loading the graph operands as an adjacency list via primary and secondary indices, and the other one for returning the graph join result (§3). We implemented a graph equi-join operator for both edge semantics (§4). We show that bulk graphs could be also used to efficiently maintain positive incremental updates for the graph conjunctive join (§5). After outlining the formal computational complexity of both graph join semantics as implemented in our algorithm (§6), our benchmarks (§7) clearly show that the combination of efficient algorithms and an ad-hoc data structure outperform the graph join definition over current relational databases (PostgreSQL) in SQL and on graph query languages for both property graphs and RDF data (Cypher over Neo4J and SPARQL over Virtuoso). Similar conclusions can be done for incremental updates.

2 LOGICAL DATA MODEL

The term *property graph* usually refers to a directed, labelled and attributed multigraph. Regarding Figure 1, we associate a collection of *labels* to every vertex and edge (e.g., {User} or {Cites}). Further on,

vertices and edges may have arbitrary named attributes (*properties*) in the form of key-value pairs (e.g., {TITLE=Graphs, 1AUTHOR=Alice}). Property-value associations of vertices and edges can be represented by relational tuples; this is a common approach in literature even when graphs have no fixed schema. Relational tuples with no conflicting key-value pairs can be merged with the \oplus operator. As RDF graphs can be always represented in Property Graphs, we use the latter as an underlying logical model of choice for our physical model and associated algorithm. Please refer to [2, §II] for additional details.

Due to the lack of space, we moved the formal definitions of graph joins alongside the definition of both conjunctive and disjunctive semantics [2, §III]. An operational definition is given in §4 where we describe how to compute such operator on top of the physical data model. In the same technical report, we show that our formal model of choice allows graph joins to be commutative and associative operations, thus showing similar properties to relational joins and enabling the definition of multi-graph equi-joins.

3 PHYSICAL DATA MODEL

Near-Data Processing approaches for Big Data on SSDs [12, 22] provide efficient resource utilization in runtime for both read (input) and write (output) operations. To meet efficiency on reading operations, we need to exploit the principle of locality; this requires the usage of linear data structures, thus achieving sequential locality. As a consequence, we’re going to represent each graph operand and its associated primary and secondary indices as linear data structures. Efficiency on write operations is obtained by representing a graph adjacency list as fixed-size records; given that this data structure contains limited information (i.e., vertices and edges represented with their ids) that is going to be later on used for incremental updates, we call this data structure “bulk graph”.

Read Operations: Indexed Graphs Each graph operand is represented using three data structures, whose definition are represented in Figure 2b: *VertexVals* represents the adjacency list record associating each vertex to its outgoing vertices; each record has a variable size, and it is sorted by vertex hash, thus inducing a vertex bucketing. Such buckets may be accessed in constant time using a primary index, *HashOffset* of fixed-size records. After associating a unique id to each vertex $v_i \in V$, such vertices can be randomly accessed in constant time by using the secondary index *VertexIndex*.

VertexVals stores vertices alongside with their adjacency list, where the outgoing vertices are sorted by hash value and are represented by their unique *id* and *hash* value to avoid data replications. Each vertex in *VertexVals* is stored by omitting the vertices’ attribute names and serialising only the associated values ($val[1] \dots val[M]$). The graph equi-join algorithm is going to explicitly rely upon a vertex bucketing induced by the selection predicate θ , as well as each vertex outgoing edges on the target vertices. Therefore, we can achieve the advantages of a sorted hash join iff. the to-be-matched vertices are sorted by hash value. If such hash-sorted vertices are stored in a linear data structure as in Figure 2b (e.g. an array, *VertexVals*), then we could create a hash index *HashOffset* for all the vertices’ join buckets: such index is composed of fixed-size records providing both the bucket *value* and the *offset* to the first vertex of

¹<https://github.com/gyankos/graphjoin>

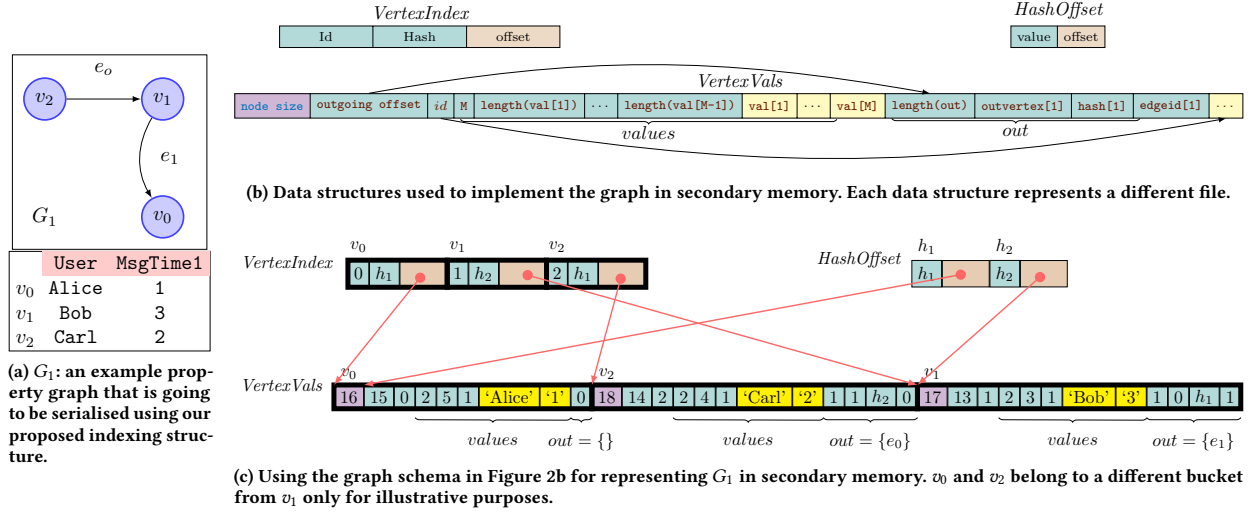
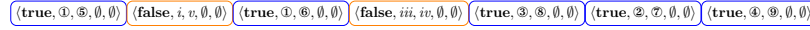
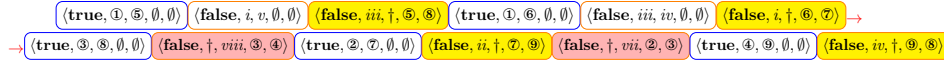


Figure 2: Indexed Graph in secondary memory.

Figure 3: Representing the bulk graph for the join $\text{Researcher}^{\text{Name}=1\text{Author}}$ Citation depicted in Figure 1f.Figure 4: Representing the bulk graph for the join $\text{Researcher}^{\text{Name}=1\text{Author}}$ Citation depicted in Figure 1g.

the bucket stored in *VertexVals*. Given that the vertices in *VertexVals* have variable size, we define a secondary index named *VertexIndex* of fixed-size records, allowing a random accessing to the vertices stored in *VertexVals* in $O(1)$ time: each record is ordered by vertex *id*, has the vertex *hash* and contains the *offset* to where the vertex data is stored in *VertexVals*. Figure 2c depicts a serialised representation of the graph in Figure 2a: all the labels and the edge values are not serialised but are still accessible via *id*. We access our three data structures using memory mapping: the operating system directly handles which pages need to be either cached or retrieved from secondary memory, thus enabling opaque caching.

Write Operations: Bulk Graphs The result of the graph join algorithm could be too large to fit in the main memory. For this reason, some implementations like Virtuoso and PostgreSQL do not explicitly store the result unless explicitly required but evaluate the query step by step. On the other hand, Neo4j sometimes fails to provide the final result due to the employment of all the available primary memory (Out of Primary Memory, **OOM1**). To store the whole result as fixed-size blocks in secondary memory, we chose to implement an ad-hoc graph data structure represented as an adjacency list (Figure 3), where each entry represents a result's vertex (in blue, **isVertex=true**) and, possibly, its adjacent edges (in orange, **isVertex=false**). So, vertices and adjacent edges are differentiated by an **isVertex** boolean flag, so that each block $\langle \text{isVertex}, \text{Left}, \text{Right}, dt, 0 \rangle$ represents either a vertex or an edge

having an id $dt(\text{Left}, \text{Right})$, where Left (Right) comes from the left (right) operand and dt is a bijection mapping two ids to one single id. \emptyset slots are only used for disjunctive semantics (Figure 4): when an edge coming from one operand is returned, a placeholder \dagger stands for the edge missing from the other operand, and \emptyset slots are filled with additional vertex information for reaching the destination node in the resulting graph. The information associated with each bulk graph vertex and the associated outgoing edges can be reconstructed while visiting the bulk graph by accessing both operands' *VertexIndex*. §5 will show that, given a set of (e.g., two) bulk graphs, we can reconstruct an indexed graph.

4 GRAPH EQUI-JOIN ALGORITHM

Given that the most common use of join operations only includes equality predicates among different tuples [8], we focus on an algorithm for equi-join predicates θ . This paper extends our previous work [3] by providing a novel join algorithm that performs both conjunctive and disjunctive semantics.

The GRAPH EQUI-JOIN ALGORITHM in Algorithm 1 consists of four main parts: generating the hashing function from the vertex θ predicate (Line 40), loading the graph operand G_x in primary memory as a red-black tree map_x and associating vertex id a hash value (*bucket*) alongside the associated outgoing vertices (Line 41), creating the secondary memory indices depicted in the previous section

from the ordered map data structure (map_x^2 , Line 42), and then executing a partition sorted join over the adjacency list representation of the serialised operands (Line 44).

First, we infer the hashing function h from θ : if $\theta(u, v)$ is a binary predicate between distinct attributes from u and v , h is defined as a linear combination of hash functions over the attributes of either u or v . We implemented in C++ the Java hashing for lists as $h(\vec{x}) = 3 \cdot 17^{|\vec{x}|} + \sum_{i=1}^{|\vec{x}|} 17^i \cdot h(\vec{x}_i)$, where h is a hashing function for each possible value \vec{x}_i . If no function could be inferred from θ , h is a constant function returning a default hashing value (e.g., 3).

LOADING performs a vertex bucketing for graph operand G_x in main memory: its outcome is a vertex set stored as an ordered multi-map map_x implemented over red-black trees, where each vertex v is stored in a collection $map_x[h(v)]$, where h is the aforementioned hashing function. Before doing so, v 's adjacency list $v.out$ is sorted by target vertex hash value.

INDEXING stores the map_x in secondary memory as the read graph in §3: as the vertices are sorted by hash value via the ordered multi-map, the serialization is sequential and straightforward. Please observe that the outgoing edges are sorted by the target node's hash.

The last step performs the actual conjunctive join over the serialised graph (JOIN); let us now discuss the conjunctive join semantics ($es \equiv \wedge$): both operands' data structures are accessed from secondary memory through memory mapping. Line 43 provides the buckets' intersection via *HashOffset*: while performing a linear scan over the sorted buckets in such primary index, the iteration checks if both operands have a bucket with the same hash value k is extracted. When this happens, the two buckets can be accessed jointly (Line 15). If there are two distinct vertices v and v' , one for each operand, satisfying θ , a newly joined vertex is created by merging those as $v \oplus v'$ (Line 17). Next, unlike the relational join, we also visit the adjacent vertices for both operands. Similarly to Line 43, the hash-sorted edges e and e' having v and v' as source nodes induce a bucketing (Line 23 and 24), and for each hash-matching edge bucket, we check if the destination vertices ($e.dst, e'.dst$) meet the join conditions alongside with the to-be-joined edges (Line 31). Please note that edges are not filtered by θ predicate, but only combined according to specific semantics. The design of the previously described sorted bucket-based scan also permits the reduction of the number of page faults, since all the vertices with the same hash value are always stored in a contiguous block in *VertexVals*. The resulting graph is stored in a bulk graph (Figure 3) where only the vertices *id* from the two graph operators appear as pairs. We can show that the algorithm performs a Cartesian product over both vertex sets and both edge sets in the worst-case scenario and performs in quasi-linear time when a "good" hashing function is found, therefore showing a good theoretical time complexity:

LEMMA 4.1. *Given two graph operands G_a and G_b and a θ binary operator, the conjunctive algorithm runs in time $T_\wedge(G_a, G_b) \in O(|\mathcal{V}(a)||\mathcal{V}(b)||\mathcal{E}(a)||\mathcal{E}(b)|)$ in the worst-case scenario, and is $T_\wedge(G_a, G_b) \in O(|\mathcal{E}(a) \cup \mathcal{E}(b)| + |\mathcal{V}(a)| \log |\mathcal{V}(a)| + |\mathcal{V}(b)| \log |\mathcal{V}(b)|)$ in the best case scenario (proof at [2, §IV]).*

When $es \equiv \vee$, the algorithm runs with the disjunctive semantics: all the edges discarded from the intersection for $v \oplus v'$ within the conjunctive semantics at Line 30 should be now considered (Line 33),

Algorithm 1 Graph Equi-Join Algorithm

Input: Two graph operands G_a, G_b , an Equi-Join θ , and **es** semantics

Output: $G_a \bowtie_{\theta}^{es} G_b$

Global: $h, BI, map_a^2, map_b^2, \theta$

```

function DISJUNCTION( $u, u', E_L, E_R, k', BulkGraph$ )
2:  if  $k' \in BI$  then
      for  $e \in E_L$  and  $v' \in map_b^2[k']$  do
4:    if  $\theta(e.dst, v')$  then
       $\epsilon := \text{new Edge}(u', v')$ 
6:       $BulkGraph.E \leftarrow e \oplus \epsilon$ 
      for  $v \in map_a^2[k']$  and  $e \in E_R$  do
8:        if  $\theta(v, e.dst)$  then
           $\epsilon := \text{new Edge}(u, v)$ 
10:          $BulkGraph.E \leftarrow \epsilon \oplus e$ 

12: function JOIN( $map_a^2, map_b^2, isDisj$ )
       $BulkGraph = \text{open\_write\_file}$ 
14:   for each bucket  $k \in BI$  do
      for  $(v, v') \in map_a^2[k] \cdot map_b^2[k]$  do
16:     if  $\theta(v, v')$  then
       $u := v \oplus v'; BulkGraph.V \leftarrow u$ 
18:      $U := \emptyset$ 
       $I := \text{Keys}(v.map) \cap \text{Keys}(v'.map)$ 
20:     if  $isDisj$  then
       $U := \text{Keys}(v.map) \cup \text{Keys}(v'.map)$ 
22:     else
       $U := I$ 
24:     for  $k' \in U$  do
      if  $isDisj$  then
26:        $E_L := \text{EdgeSet}(v.map[k'])$ 
        $E_R := \text{EdgeSet}(v'.map[k'])$ 
28:       if  $\neg isDisj \vee k' \in I$  then
         for  $(e, e') \in v.map[k'] \cdot v'.map[k']$  do
30:           if  $\theta(e.dst, e'.dst)$  then
              $BulkGraph.E \leftarrow e \oplus e'$ 
32:             if  $isDisj$  then  $E_L := E_L \setminus \{e\}; E_R := E_R \setminus \{e'\};$ 
             if  $isDisj$  then DISJUNCTION( $v, v', E_L, E_R, k', Result$ )
34:       else
         DISJUNCTION( $v, v', E_L, E_R, k', Result$ )
36:   return  $BulkGraph$ 

38: function GEJA( $G_a, G_b, \theta, es$ )
      Setting  $\theta$  as a global parameter
40:    $h \leftarrow H(\theta, A_v, A_b)$ 
       $map_a \leftarrow \text{Loading}(h, G_a); map_b \leftarrow \text{Loading}(h, G_b)$ 
42:    $map_a^2 \leftarrow \text{Indexing}(map_a); map_b^2 \leftarrow \text{Indexing}(map_b);$ 
       $BI := \text{Keys}(map_a^2) \cap \text{Keys}(map_b^2)$ 
44:   return JOIN( $map_a^2, map_b^2, es \equiv \vee$ );

```

Algorithm 2 Graph Conjunctive Equi-Join With Incremental Update

Input: Operands $(G_R \cup G_R^+)$ G_S and an Equi-Join θ

Output: $(G_R \cup G_R^+) \bowtie_{\theta}^{\vee} G_S$

Global: $BI, map_a^2, map_b^2, \theta$

```

function CONJUNCTIVEINCRUPD( $G_R \cup G_R^+, G_S, \theta$ )
2:   $h \leftarrow H(\theta, A_v, A_b)$ 
       $\triangleright$  GJEA: creating the materialized view and global initialization
4:   $M \leftarrow \text{GJEA}(G_R, G_S, \theta, \wedge)$ 
       $Load \leftarrow \text{Loading}(h, G_R^+); Idx \leftarrow \text{Indexing}(Load)$ 
6:   $BI' := \text{Keys}(Load) \cap \text{Keys}(map_b^2)$ 
       $Join := \text{Join}(Idx, map_b^2, BI', \wedge)$ 
8:  return Loading2( $x \mapsto 0, M \cup Join$ )

```

if they come either from the left operand (Line 3) or from the right one (Line 7). E_L and E_R sets contain the aforementioned discarded

edges, that are then going to be considered by the **DISJUNCTIVE** function. Among all such edges, we can run our disjunctive join first over the ones coming from the left operand (Line 3): since the final edges must only connect vertices belonging to the final vertex set, we consider only those e that have a destination vertex “ $e.dst$ ” which hash value appears in *HashIndex* (Line 2). Moreover, it has to satisfy the binary predicate θ jointly with another vertex v' coming from the opposite operand (Line 4). Hence, we establish an edge $e \oplus \varepsilon$ having the same values and attributes of e and the same set of labels; then, such edge is written in the bulk graph (Lines 5 and 6). Similar considerations should be done by the edges from the right operand (Lines 7-10).

As we can see from **DISJUNCTION**, the disjunctive semantics potentially increases the final number of the edges while keeping the vertex set size unchanged, as requested (proof at [2, §IV]):

LEMMA 4.2. *With respect to the time complexity, the best (worst) case scenario of the disjunctive semantics is asymptotically equivalent to the conjunctive semantics in its best (worst) case scenario, i.e. $\limsup_{(|G_a|, |G_b|) \rightarrow (+\infty, +\infty)} \frac{T_A(|G_a|, |G_b|)}{T_V(G_a, G_b)} = 1$, under the same algorithmic conditions, where $|G_x|$ is a shorthand for $|V(x)| + |E(x)|$.*

5 INCREMENTAL UPDATES

Current literature defines graph incremental queries only over graph traversal queries returning a subgraph of the original graph operand, and do not involve the creation of novel graphs as an output [25]. Similarly, graph query caching systems such as [31] mainly involve graph pattern matching queries where no graph transformations are involved. We then propose a positive incremental update algorithm for graph conjunctive joins.

To motivate the adoption of such a general approach to all the current graph databases, we want to show that graph join operations can be implemented alongside materialised views. This requirement is crucial for Big Data as it guarantees that, when graphs are updated within the database, we don’t need to recompute the join operation from scratch. To meet this goal, we implement graph Equi-Joins’ incremental updates similarly to how materialized views are updated in relational databases (RDBMS). Let us briefly state how those refresh materialized views over binary joins work: let us suppose to have two relationships, R and S , where tuples are added (R^+, S^+) or removed (R^-, S^-). We can now express the final updated relationships as $R \cup R^+ \setminus R^-$ and $S \cup S^+ \setminus S^-$ respectively. Consequently, if we want to update the materialised view $M = R \bowtie_{\theta} S$ then intuitively² we need to (i) remove the join between the deleted tuples from the (previously computed) materialised view, (ii) join the tuples added in the second operand with the left operand minus the tuples removed from it (iii) and vice-versa, (iv) join only the newly added tuples together, and finally (v) update M as the union of all the previous computations. This equivalence is known at least from the late nineties [30]. In this current paper, we investigate the positive incremental updates, i.e. when data is added and not removed from the operands ($R^- = S^- = \emptyset$). If we also narrow down the problem on updating just one operand (e.g., the left one $R^+ \neq \emptyset$, and $S^+ = \emptyset$), then the whole problem boils down to computing $M \cup (R^+ \bowtie_{\theta} S)$. This requires extending the materialized view M

²More formally, $(M \setminus (R^- \bowtie_{\theta} S^-)) \cup ((R \setminus R^-) \bowtie_{\theta} S^+) \cup (R^+ \bowtie_{\theta} (S \setminus S^-)) \cup (R^+ \bowtie_{\theta} S^+)$

by θ -joining the left increment with the whole right operand. As a consequence, instead of re-indexing the whole left operand, that is $R \cup R^+$, we can index only R^+ before performing the theta join of such operator against R . We can formally prove that similar results can be directly mimicked by the graph conjunctive join, where relationships in the former equation are directly replaced by graph operands.

Our paper’s bibliography motivates this restriction for temporal social network scenarios, where we can assume – as some other authors before us [16, 34] – that novel information for Big Data is always added and never removed. Within this paper, we’re only going to consider the positive incremental updates for the conjunctive semantics (Algorithm 2). Albeit this use case might seem too narrow, we must remind the reader that, as shown in the introduction, the conjunctive semantics appears already as an essential operation for many different relevant use cases, as well as showing that we can reassemble all the primary graph routines from Algorithm 1 for expressing a novel operation, the Graph Conjunctive Equi-Join with Incremental Updates. Even in this case, our goal will be to show that such novel operations are going to outperform refresh operations over relational databases, given that current graph databases haven’t query update operations as such (§7).

After creating the bulk graph for the intermediate representation $M := G_R \bowtie_{\theta} G_S$ (Line 4), we want to update such result by considering the increment G_R^+ . As it happens in relational databases, we load just G_R^+ and then index it (Line 5) and then run a slightly different version of the **JOIN** algorithm (Line 7): in fact, we must consider that the first vertex in G_R^+ does not necessarily start from index 0, and we must take into account that not all the vertices are represented. Then, we need to take the two bulk graphs and create a materialized view out of them, which is an indexed graph. For graphs, graph unions can be implemented as the vertex and edge set unions [15]: given that the bulk graphs are just an adjacency list, we need to scan the two files and merge the two adjacency lists in primary memory (**MUJoin**). We save the final result as an Indexed Graph via a slight edit of the Loading phase (**Loading₂**) where we both combine the bulk graph vertices (u_i, v_j) as one single vertex $u_i \oplus v_j$, and we create an indexed graph not from secondary memory data, but from the primary-memory loaded bulk graph (Line 8).

6 RELATED WORK

Discrete Mathematics. Every graph product of two graphs [13] produces a graph whose vertex set is the cross product of the operands’ vertex sets, thus creating a set of vertex pairs; the edge set computation changes according to the different graph product definition. Please observe that vertex pairs differ from the relational algebra’s cartesian product outcome, where the two vertices are *combined* via \oplus . Nevertheless, such operations do not take into account the naming of the resulting table, which in graph databases could be assimilated to the vertices’ (and edges’) labels. [33] provided a Kronecker Product for edge uncertainty, even though no information on how to combine the data associated with either vertices or edges is given: the authors implemented a matrix operation within a relational database. Conjunctive semantics implements Kronecker Products for graph joins. Last, the graph conjunctive

equi-joins of two node-labelled automata A and B provides a new automaton C , where vertices $u \oplus v$ are initial (or accepting) states iff. both u and v are initial (or accepting) states in both operands. By denoting $\mathcal{L}[C]$ as the language generated by an automaton C , we can show that $\mathcal{L}[C] = \mathcal{L}[A] \cap \mathcal{L}[B]$ [1].

Graph Similarity. Kronecker graph products have also a more practical application in the computation of **graph kernels**. Graph kernels allow mapping graph data structures to feature spaces (usually a Hilbert space in \mathbb{R}^n for $n \in \mathbb{N}_{>0}$) [23] so to express graph similarity functions that can then be adopted for both classification [28] and clustering algorithms. Kronecker (graph) products were used first to generate one single directed graph, over which the summation of the weighted walks provides the desired graph kernel result [23]. Kronecker graph products are also used in current literature to generate the minimal general generalization of two graphs. Such an intermediate graph is then used within a graph edit-distance [6] that can be expressed as a kernel function because it is a proper metric. Given that all such techniques require a Kronecker graph product that can be expressed via a conjunctive join, our conjunctive join algorithm reveals to be beneficial for efficiently computing those metrics over big graphs.

Schema alignment, data similarity and integration techniques relying on the similarity flooding algorithm [21] require the creation of a “pairwise connectivity graph” (PCG) from two different graphs (e.g., graph schemas), namely A and B . Such a graph is the outcome of a conjunctive graph join:

$$((u \oplus v), \text{label}, (u' \oplus v')) \in E_{PCG} \Leftrightarrow (u, \text{label}, u') \in E_A \wedge (v, \text{label}, v') \in E_B$$

Given that the same algorithm could be also applied to a variety of different scenarios such as (graph) schema alignment, matching semistructured schemas with instance data as well as finding similarities among data instances via self graph joins [21], an efficient implementation of the graph conjunctive algorithm will be also beneficial to these disparate scenarios.

Please note that none of the aforementioned approaches provided an efficient implementation of the required graph product, as the graph product was just a pre-processing step required to solve the main problem and not the target of the research question.

Relational Databases. Walking in the footsteps of current literature, we can represent each graph operand as two distinct tables, one listing the vertices and the other listing the edges [2, Appendix C] In addition to the attributes within the vertices’ and the edges’ tables, we assume that each row (on both vertices and edges) has an attribute *id* enumerating vertices and edges. Concerning SQL interpretation of such graph join, we first join the vertices. The edges are computed through the join query provided in [2, Appendix C]: the root and the leaves are the results of the θ join between the vertices, while the edges appear as the intermediate nodes. An adjacency list representation of a graph, like the one proposed in the current paper, reduces the joins within the relation solution to one (each vertex and edge is traversed only once), thus reducing the number of required operation to create the resulting graph. On the other hand, graphs as secondary memory adjacency

lists enable efficient distributed algorithms for graph traversals [18].

Graph Query Languages. Different graph query languages³ subsume different graph database management system architectures: while SPARQL allows access to multiple graphs resources via *named graphs*, Cypher stores only one graph per database at a time, and hence do not naturally support binary graph join operations. Different graph operands might be still expressed via distinct connected components that are potentially associated with different labels. Graph query languages such as Cypher and SPARQL do not provide a specific keyword to compute the graph join between two graphs. As a consequence, the graph join must be written as an explicit query composed of several different clauses. Concerning Cypher, the **CREATE** clause has to be used to generate new vertices and edges from graph patterns extracted through the **MATCH...WHERE** clause and intermediate results are merged with **UNION ALL**. By analyzing the associated query plan, we observed that the query language requires us to traverse the same graph five times, thus visiting the same graph database and all the associated sub-patterns multiple times. On the other hand, SPARQL reduces the number of the distinct traversed patterns to two, but the need of invoking an **OPTIONAL** clause makes the overall query plan inefficient because all the certain paths are left joined with the optional ones. At this point, the **CONSTRUCT** clause is required if we want to finally combine the traversed paths from both graphs into a resulting graph. The construction of new graphs is not part of the algebraic optimisation of the SPARQL queries, which are originally designed to return tables; this introduces additional computational overhead. This limitation is also shared with Cypher.

7 EXPERIMENTAL RESULTS

Our experiments were performed on top of two datasets⁴: we used a real social network dataset (Friendster [32]) for simulating a sparse graph, and we used the SNAP Kronecker graph generator *krongen* for simulating real-world transport and communication networks with heavy-tailed degree distributions [19]. Given that both datasets do not come with vertex value information, we enriched them using the guidelines of the LDBC Social Network Benchmark protocol. We associated each vertex (user-id) a name, surname, e-mail address, organization name, year of employment, sex, and city of residence.

We performed our tests over a Lenovo ThinkPad P51 with a 3.00 GHz (up to 4.00 GHz) Intel Xeon processor and 64 GB of RAM up to 4.000 MHz. The tests were performed over an SSD with an ext4 (GNU/Linux) File System with a free disk space of 50GB. We evaluate the graph join using as operands two distinct sampled sub-graphs with the same vertex size ($|V|$), where the θ predicate is the following one: $\theta(u, v) \stackrel{\text{def}}{=} u.\text{Year1} = v.\text{Year2} \wedge u.\text{Organization1} = v.\text{Organization2}$. Such predicate does not perform a perfect 1-to-1 match with the graph vertices, thus allowing to test the algorithm with different multiplicities values. We performed the same join query over the two different datasets for both semantics.

³Please refer to [2, Appendix B] for an in-depth explanation of the limitations of such languages. In this technical report, we also provide the original benchmark queries for reproducibility purposes.

⁴<https://osf.io/xney5/>

We used the default configurations for **Neo4J 3.5.0** and **PostgreSQL 10.6**, while we changed the cache buffer configurations for **Virtuoso 7.20.3230** (as suggested in the configuration file) for 64 GB of RAM; we also kept the default multi-threaded query execution plan. PostgreSQL queries were evaluated through the `psql` client and benchmarked using both `explain analyse` and `\timing` commands; the former allows to analyse SQL's query plans. Virtuoso was benchmarked by using `unixODBC` connections; SPARQL's associated query plan was analysed via Virtuoso's `profile` statement. Cypher queries were sent using the Java API but the graph join operation was performed only in Cypher through the `execute` method of an `GraphDatabaseService` object.

GRAPH EQUI-JOIN. For each distinct dataset, we generate two graphs collections for both the left and the right operand. Each collection was generated by random walk sampling each given dataset starting from the same vertex, and storing each graph once of a number of the traversed vertices is a power of 10, from 10 to 10^8 . The only parameter that was changed for generating distinct collections for the left and right operand's collections was the graph traversal seed. Overall, the sampling approach guarantees that each left and right operand subgraph always share at least one single vertex and that each operand approximates the graph degree distribution of the original data set. Furthermore, each left (right) operand of size 10^i is always a subgraph of all the operands 10^{i+n} for $n > 1$ and $i + n \leq 8$. For Friendster, we started the random walk from one of the hub nodes, while we randomly picked it for the other dataset.

The experiments show that our approach outperforms both graph equi-join semantics on different current query languages interpretation for both graph and relational databases. We consider the time to (i) serialise our data structure (Loading) and (ii) evaluate the query plan (Indexing and Joining). This twofold analysis is required because, in some cases, the costly creation of several indices may lead to better query performance. Given that our graph join result is represented as a bulk graph where no vertex and edge information is serialised, all the queries fed to our competitors are designed only to return the vertex and the edge id that are matched. The lack of ancillary data attached to either vertices or edges allows better comparison of query evaluation times, which are now independent of the values' representations and tailored to evaluate both the access time required for the loaded operator and returning the joined graph.

Table 1 provides a comparison for the actual loading, indexing and joining running time for our conjunctive and disjunctive graph join implementation. This evaluation confirms that the actual conjunctive joining time is negligible, while the whole computational burden is carried out by both the operand loading and the indexing. On the other hand, the disjunctive join task dominates over the loading phase for bigger graphs. Moreover, more dense graphs are affected more by the indexing time than by the loading time. The experimental evaluation also confirms that the disjunctive semantics contains a superset of operations with respect to the conjunctive one, and therefore its computation takes more time than the disjunctive one. We also analysed the main memory consumed by our algorithm using `malloc_stats` and the Ubuntu System Monitor, and we might observe that our algorithm might take at most 22GB of primary memory.

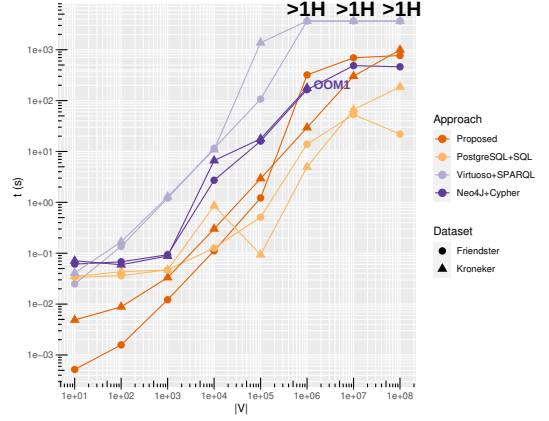


Figure 5: Loading time over the enriched Friendster and Kronecker Datasets. $1H = 3.60 \cdot 10^3$ s

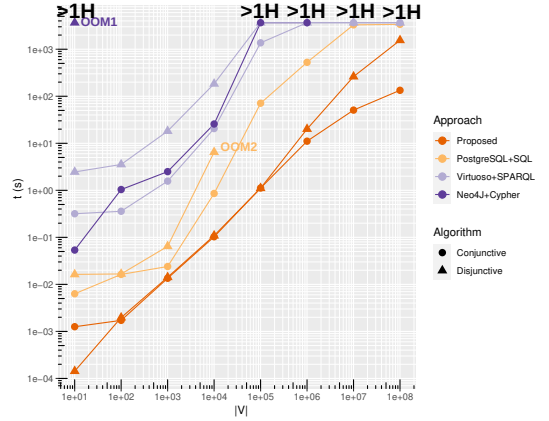


Figure 6: Graph Equi-Joins over the enriched Friendster Dataset: (Indexing and) Join Time. We remark the benchmarks that exceeded the one hour threshold with $>1H$, as well as the algorithms failing to compute due to either primary (OOM1) or secondary (OOM2) out of memory error.

Figure 5 [2, Table I] compares the loading time for both datasets for our solution as well as for all the competitors. Please note that the loading time in PostgreSQL corresponds to loading the vertices' and edges' tables for both operands, in Virtuoso it corresponds to the creation of two distinct named graphs and in Neo4J corresponds to the creation of two distinct connected components within one single graph database. Globally, graph databases are less performant than relational databases: concerning the loading phase, Virtuoso creates triple indices for fastening graph traversal operations, while in Neo4J an edge creation always requires the checking whether two vertices already exist. PostgreSQL is not affected by this computational burden, given that the graph representation is tabular. Given that our proposed solution also requires linking vertices with edges, PostgreSQL outperforms our solution for loading big operands while our implementation is faster for smaller datasets.

Table 1: Graph Conjunctive and Disjunctive Equi-Join: Separating Loading, Indexing and Joining time for the graph join algorithm, thus comparing the theoretical computational complexity and the empirical evaluation. Please note that the Loading and Indexing time are the same for both semantics.

$ V_1 = V_2 $	Friendster				Kronecker Graph			
	Loading (s)	Indexing (s)	Conj. Join (s)	Disj. Join (s)	Loading (s)	Indexing (s)	Conj. Join (s)	Disj. Join (s)
10^1	$5.20 \cdot 10^{-4}$	$1.19 \cdot 10^{-3}$	$7.00 \cdot 10^{-5}$	$2.42 \cdot 10^{-4}$	$4.88 \cdot 10^{-3}$	$2.44 \cdot 10^{-4}$	$8.70 \cdot 10^{-5}$	$1.00 \cdot 10^{-4}$
10^2	$1.58 \cdot 10^{-3}$	$1.62 \cdot 10^{-3}$	$1.17 \cdot 10^{-4}$	$3.51 \cdot 10^{-4}$	$8.84 \cdot 10^{-3}$	$8.16 \cdot 10^{-4}$	$2.21 \cdot 10^{-4}$	$2.85 \cdot 10^{-4}$
10^3	$1.22 \cdot 10^{-2}$	$1.30 \cdot 10^{-2}$	$4.80 \cdot 10^{-4}$	$1.25 \cdot 10^{-3}$	$3.33 \cdot 10^{-2}$	$7.11 \cdot 10^{-3}$	$1.06 \cdot 10^{-3}$	$9.57 \cdot 10^{-4}$
10^4	$1.12 \cdot 10^{-1}$	$1.01 \cdot 10^{-1}$	$2.08 \cdot 10^{-3}$	$8.14 \cdot 10^{-3}$	$3.01 \cdot 10^{-1}$	$7.26 \cdot 10^{-2}$	$8.66 \cdot 10^{-3}$	$4.47 \cdot 10^{-3}$
10^5	$1.22 \cdot 10^0$	$1.10 \cdot 10^0$	$2.31 \cdot 10^{-2}$	$1.34 \cdot 10^{-2}$	$2.95 \cdot 10^0$	$8.67 \cdot 10^{-1}$	$6.53 \cdot 10^{-2}$	$5.25 \cdot 10^{-2}$
10^6	$3.20 \cdot 10^2$	$1.09 \cdot 10^1$	$2.15 \cdot 10^{-1}$	$9.72 \cdot 10^0$	$2.95 \cdot 10^1$	$9.89 \cdot 10^1$	$6.52 \cdot 10^{-1}$	$2.23 \cdot 10^0$
10^7	$6.93 \cdot 10^2$	$4.99 \cdot 10^1$	$9.27 \cdot 10^{-1}$	$2.12 \cdot 10^2$	$3.02 \cdot 10^2$	$1.15 \cdot 10^2$	$7.05 \cdot 10^0$	$4.06 \cdot 10^2$
10^8	$7.69 \cdot 10^2$	$1.32 \cdot 10^2$	$2.15 \cdot 10^0$	$1.42 \cdot 10^3$	$9.93 \cdot 10^2$	$8.74 \cdot 10^2$	$2.20 \cdot 10^1$	$> 1H$

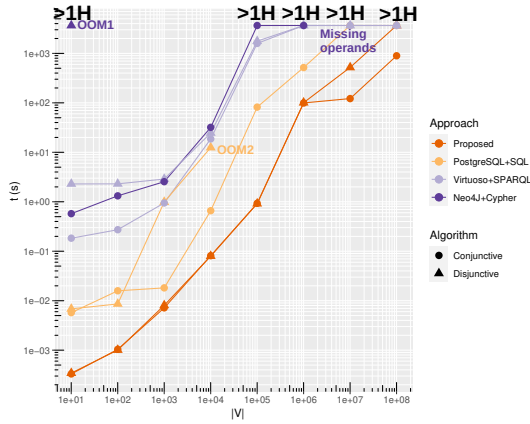


Figure 7: Graph Equi-Joins over the enriched Kronecker Graph Dataset: (Indexing and) Join Time.

Figure 6 [2, Table II] and 7 [2, Table III] compare the conjunctive and disjunctive implementation to the competitor’s solutions for the Friendster (and Kronecker) dataset. For both PostgreSQL and Neo4J, the (join) indexing happens during the query evaluation, and therefore we compare the sum of our indexing and joining time to the competitors’ query evaluation. On the other hand, our proposed indexing and join time always outperforms the competitors’ query plans by at least one order of magnitude, even when the join algorithm cost is considered as the sum of both loading and indexing+joining. Please also note that the Virtuoso query engine rewrites the SPARQL query into SQL and, as a result of this, two SQL queries were performed in both Virtuoso and PostgreSQL.

The disjunctive Equi-Join algorithm always runs faster than the best performer (PostgreSQL)’s conjunctive Equi-Join, thus remarking the effectiveness of our implementation independently from the semantics of choice. Furthermore, all the implementations of the disjunctive semantics suffer from lower performances as expected; the reason is twofold: both the for-all statement requires for SQL and Cypher an aggregation that has already been proved to be inefficient [4], jointly with an increase of the patterns that need to be visited and that the current languages’ query plan does not allow

to incorporate as one single algorithm. Last, our implementation also shows to be more memory efficient if compared with other implementations: producing the bulk graph for the 10^8 operands from the Kronecker dataset takes more than 1 H after writing more than 50GB in secondary memory while producing the same bulk graph for 10^7 operands required only 30GB of free disk space. On the other hand, PostgreSQL consumed all the remaining secondary memory already with the 10^5 operands.

As a result, the inefficiency of the graph query languages is due to both the need to repeatedly traverse the same data over multiple patterns with no query plan optimisation and to the fact that such query languages are not optimised to return one graph as a result. On the other hand, the SQL interpretation of the graph conjunctive equi-join reduces the number of comparisons, thus resulting in more performative than the other relational databases. Still, our proposed solution of representing a graph as an indexed adjacency list reduces the overall number of comparisons and hence reduces the computation time of at least one order of magnitude over bigger datasets.

INCREMENTAL UPDATES. Within this paper, we consider positive incremental updates for graph conjunctive Equi-Join. We chose the biggest operands of the Kronecker Graph Dataset where all the competitors took less than one hour to compute (10^4 for the graph conjunctive join) as the basic input for $M = R \bowtie_{\theta} S$. Then, only for the left operand, we extend the former series via random graph samples with additional $10^4 + x \cdot 10^3$ operands for $x \in \{1, 2, 5, 7, 9\}$ such that the difference of graph $10^4 + x \cdot 10^3$ with graph 10^4 will become our R^+ .

Given that Neo4J (and Virtuoso) does not support creating materialized views over Cypher (and SPARQL) queries and considering that PostgreSQL is the fastest competitor, we restrict our analysis on PostgreSQL. After loading the main operands and storing their graph join as two materialized views (one for the resulting vertices and the other for the resulting edges), we load only R^+ (Copy) and we run **REFRESH MATERIALIZED VIEW** for both the vertex and the edge table representing the graph conjunctive Equi-Join result. Similarly, for benchmarking our operator we provide the following operations: after computing $M = R \bowtie S$ with the usual graph conjunctive join algorithm, we only load the R^+ graph (Load), then we run the indexing and joining algorithm for $R^+ \bowtie S$ (Idx+Join), and

Table 2: Graph Conjunctive Join and Bulk Merging vs. PostgreSQL’s Materialized Views

$ R^+ / R $	Proposed		PostgreSQL	
	Load (s)	Idx+Join+Mat (s)	Copy (s)	Refresh (s)
10%	$2.39 \cdot 10^{-2}$	$3.43 \cdot 10^{-3}$	$1.50 \cdot 10^{-2}$	$3.26 \cdot 10^{-1}$
20%	$4.73 \cdot 10^{-2}$	$6.38 \cdot 10^{-3}$	$2.04 \cdot 10^{-2}$	$3.58 \cdot 10^{-1}$
50%	$1.16 \cdot 10^{-1}$	$1.57 \cdot 10^{-2}$	$2.42 \cdot 10^{-2}$	$4.53 \cdot 10^{-1}$
70%	$1.77 \cdot 10^{-1}$	$2.25 \cdot 10^{-2}$	$4.89 \cdot 10^{-2}$	$4.60 \cdot 10^{-1}$
90%	$1.82 \cdot 10^{-1}$	$2.80 \cdot 10^{-2}$	$5.82 \cdot 10^{-2}$	$7.87 \cdot 10^{-1}$

finally, we load the two bulk graphs in primary memory and then store it using Loading₂ (Mat). Table 2 provides the result of such evaluation: while PostgreSQL still is more performant on loading the data, our implementation providing indexed graphs from bulk graphs proves to be more efficient than the competitor, thus validating the overall bulk graph approach for jet another scenario. This outcome clearly shows that bulk graphs can be efficiently reconstructed as indexed graphs with a small overhead. Last, we showed that incremental update can be efficiently implemented without ad-hoc indexing data structures by exploiting algebraic rules.

8 CONCLUSIONS

We propose a graph equi-join algorithm for the conjunctive and disjunctive semantics, outperforming the implementations on different data representations and query languages. Future works will provide extensive benchmarks for incremental updates when tuples are either added or removed in both operands. A novel algorithm for the disjunctive semantics is also going to be provided for such incremental updates. The bucketing approach would lead to a straightforward concurrent implementation of our join algorithm, where each process might run the join over a partition of the buckets. After partitioning the graph operands using a heuristic of choice, we can also exploit the incremental update algorithm for a distributed computation of graph joins over Big Data graphs. Last, we also carried out experiments to make the conjunctive semantics support the \leq predicate in θ over one attribute having partially ordered values: since our data structure is already ordered by hash value, we could use a monotone hashing function h with respect to such attribute. Such contribution as well as the implementation of left-, full-, and materialized- join algorithms will be addressed in our future works.

REFERENCES

- [1] Christel Baier and Joost-Pieter Katoen. 2008. *Principles of Model Checking (Representation and Mind Series)*. The MIT Press.
- [2] Giacomo Bergami. 2021. A Logical Model for joining Property Graphs. <https://rebrand.ly/tech-report-joins> (2021).
- [3] Giacomo Bergami, Matteo Magnani, and Danilo Montesi. 2017. A Join Operator for Property Graphs. In *Proceedings of the Workshops of the EDBT/ICDT 2017 Joint Conference (EDBT/ICDT 2017), Venice, Italy, March 21-24, 2017*.
- [4] Giacomo Bergami, André Petermann, and Danilo Montesi. 2018. THoS: an Algorithm for Nesting Property Graphs. In *GRADES-NDA*.
- [5] Michel Beuthe, Bart Jourquin, Jean-François Geerts, and Christian Koul à Ndjang’ Ha. 2001. Freight transportation demand elasticities: a geographic multimodal transportation network analysis. *Transportation Research Part E: Logistics and Transportation Review* 37, 4 (2001), 253–266.
- [6] Horst Bunke and Kim Shearer. 1998. A graph distance metric based on the maximal common subgraph. *Pattern Recognition Letters* 19, 3-4 (1998), 255–259.
- [7] Loredana Caruccio, Giuseppe Polese, and Genoveffa Tortora. 2016. Synchronization of Queries and Views Upon Schema Evolutions: A Survey. *ACM Trans. Database Syst.* 41, 2 (2016), 9:1–9:41.
- [8] Ramez Elmasri and Shamkant B. Navathe. 2015. *Fundamentals of Database Systems* (7th ed.). Pearson.
- [9] Mikhail Galkin, Diego Collarana, Ignacio Traverso Ribón, Maria-Esther Vidal, and Sören Auer. [n.d.]. SJoin: A Semantic Join Operator to Integrate Heterogeneous RDF Graphs. In *Database and Expert Systems Applications - 28th International Conference, DEXA 2017*. 206–221.
- [10] Mikhail Galkin, Kemele M. Endris, Maribel Acosta, Diego Collarana, Maria-Esther Vidal, and Sören Auer. [n.d.]. SMJoin: A Multi-way Join Operator for SPARQL Queries. In *Proceedings of the 13th International Conference on Semantic Systems*. 104–111.
- [11] Jun Gao, Jeffrey Yu, Huida Qiu, Xiao Jiang, Tengjiao Wang, and Dongqing Yang. 2012. Holistic Top-k Simple Shortest Path Join in Graphs. *IEEE Trans. on Knowl. and Data Eng.* 24, 4 (April 2012), 665–677.
- [12] Boncheol Gu, Andre S. Yoon, Duck-Ho Bae, Insoon Jo, Jinyoung Lee, Jonghyun Yoon, Jeong-Uk Kang, Moonsang Kwon, Chanho Yoon, Sangyeun Cho, Jaehoon Jeong, and Duckhyun Chang. 2016. Biscuit: A Framework for Near-Data Processing of Big Data Workloads. In *ISCA*. 153–165.
- [13] Richard Hammack, Wilfried Imrich, and Sandi Klavzar. 2011. *Handbook of Product Graphs, Second Edition* (2nd ed.). CRC Press, Inc., Boca Raton, FL, USA.
- [14] Wilfried Imrich and Sandi Klavzar. 2000. *Product Graphs. Structure and Recognition* (2nd ed.). John Wiley & Sons, Inc., New York, NY, USA.
- [15] Martin Jungmanns, Max Kießling, Niklas Teichmann, Kevin Gómez, André Petermann, and Erhard Rahm. 2018. Declarative and distributed graph analytics with GRADOOP. *PVLDB* 11, 12 (2018), 2006–2009.
- [16] Miray Kas, Matthew Wachs, Kathleen M. Carley, and L. Richard Carley. 2013. Incremental algorithm for updating betweenness centrality in dynamically growing networks. In *ASONAM*. 33–40.
- [17] Rainer Kujala, Christoffer Weckström, Richard K. Darst, Milos N. Mladenovic, and Jari Saramäki. 2018. A collection of public transport network data sets for 25 cities. *Scientific Data* 5, 1 (2018).
- [18] Alan G. Labouseur, Jeremy Birnbaum, Paul W. Olsen, Sean R. Spillane, Jayadevan Vijayan, Jeong-Hyon Hwang, and Wook-Shin Han. 2015. The G* graph database: efficiently managing large distributed dynamic graphs. *Distributed and Parallel Databases* 33, 4 (01 Dec 2015), 479–514.
- [19] Jure Leskovec, Deepayan Chakrabarti, Jon M. Kleinberg, Christos Faloutsos, and Zoubin Ghahramani. 2010. Kronecker Graphs: An Approach to Modeling Networks. *Journal of Machine Learning Research* 11 (2010), 985–1042.
- [20] József Marton, Gábor Szárnyas, and Dániel Varró. 2017. Formalising opencypher Graph Queries in Relational Algebra. *CoRR abs/1705.02844* (2017).
- [21] Sergey Melnik. 2004. *Generic Model Management: Concepts and Algorithms*. Lecture Notes in Computer Science, Vol. 2967. Springer.
- [22] Marcus Paradies. 2019. CryoDrill: Near-Data Processing in Deep and Cold Storage Hierarchies. In *CIDR*.
- [23] Nagiza F. Samatova, William Hendrix, John Jenkins, Kanchana Padmanabhan, and Arpan Chakraborty. 2013. *Practical Graph Mining with R*. Chapman & Hall/CRC.
- [24] Robert Speer and Catherine Havasi. 2013. *ConceptNet 5: A Large Semantic Network for Relational Knowledge*. Springer, Berlin, Heidelberg, 161–176.
- [25] Gábor Szárnyas. 2018. Incremental View Maintenance for Property Graph Queries. In *Proceedings of the 2018 International Conference on Management of Data (SIGMOD ’18)*. Association for Computing Machinery, New York, NY, USA, 1843–1845.
- [26] Damian Szklarczyk, John H. Morris, Helen Cook, Michael Kuhn, Stefan Wyder, Milan Simonovic, Alberto Santos, Nadezhda T. Doncheva, Alexander Roth, Peer Bork, Lars Juhl Jensen, and Christian von Mering. 2017. The STRING database in 2017: quality-controlled protein-protein association networks, made broadly accessible. *Nucleic Acids Research* 45, Database-Issue (2017), D362–D368.
- [27] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. [n.d.]. ArnetMiner: Extraction and Mining of Academic Social Networks. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (Las Vegas, Nevada, USA) (KDD ’08)*. 990–998.
- [28] Koji Tsuda and Hiroto Saigo. 2010. Graph Classification. In *Managing and Mining Graph Data*. 337–363.
- [29] Johan Ugander, Brian Karrer, Lars Backstrom, and Cameron Marlow. 2011. The Anatomy of the Facebook Social Graph. *CoRR* (2011). arXiv:1111.4503
- [30] Dimitra Vista. 1996. *Optimizing Incremental View Maintenance Expressions in Relational Databases*. Ph.D. Dissertation. University of Toronto.
- [31] Jing Wang, Nikos Ntarmos, and Peter Triantafyllou. 2017. GraphCache: A Caching System for Graph Queries. In *EDBT*. 13–24.
- [32] Jaewon Yang and Jure Leskovec. 2012. Defining and Evaluating Network Communities Based on Ground-Truth. In *ICDM*. 745–754.
- [33] Kangfei Zhao and Jeffrey Xu Yu. [n.d.]. All-in-One: Graph Processing in RDBMSs Revisited. In *Proceedings of the 2017 ACM International Conference on Management of Data*. 1165–1180.
- [34] Y. Zhou, H. Cheng, and J. X. Yu. 2010. Clustering Large Attributed Graphs: An Efficient Incremental Approach. In *ICDM*. 689–698.

An Automatic Schema-Instance Approach for Merging Multidimensional Data Warehouses

Yuzhao Yang

IRIT-CNRS (UMR 5505), Université de Toulouse
Toulouse, France
yuzhao.yang@irit.fr

Franck Ravat

IRIT-CNRS (UMR 5505), Université de Toulouse
Toulouse, France
franck.ravat@irit.fr

Jérôme Darmont

Université de Lyon, Lyon 2, UR ERIC
Lyon, France
jerome.darmont@univ-lyon2.fr

Olivier Teste

IRIT-CNRS (UMR 5505), Université de Toulouse
Toulouse, France
olivier.teste@irit.fr

ABSTRACT

Using data warehouses to analyse multidimensional data is a significant task in company decision-making. The need for analyzing data stored in different data warehouses generates the requirement of merging them into one integrated data warehouse. The data warehouse merging process is composed of two steps: matching multidimensional components and then merging them. Current approaches do not take all the particularities of multidimensional data warehouses into account, e.g., only merging schemata, but not instances; or not exploiting hierarchies nor fact tables. Thus, in this paper, we propose an automatic merging approach for star schema-modeled data warehouses that works at both the schema and instance levels. We also provide algorithms for merging hierarchies, dimensions and facts. Eventually, we implement our merging algorithms and validate them with the use of both synthetic and benchmark datasets.

CCS CONCEPTS

• Information systems → Data warehouses.

KEYWORDS

Multidimensional data warehouse, schema-instance merging, automatic integration

ACM Reference Format:

Yuzhao Yang, Jérôme Darmont, Franck Ravat, and Olivier Teste. 2021. An Automatic Schema-Instance Approach for Merging Multidimensional Data Warehouses. In *25th International Database Engineering & Applications Symposium (IDEAS 2021)*, July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3472163.3472268>

1 INTRODUCTION

Data warehouses (DWs) are widely used in companies and organizations as an important Business Intelligence (BI) tool to help build

decision support systems [9]. Data in DWs are usually modeled in a multidimensional way, which allows users to consult and analyze the aggregated data through multiple analysis axes with On-Line Analysis Processing (OLAP) [14]. In a company, various independent DWs containing some common elements and data may be built for different geographical regions or functional departments. There may also exist common elements and data between the DWs of different companies. The ability to accurately merge diverse DWs into one integrated DW is therefore considered as a major issue [8]. DW merging constitutes a promising solution to provide more opportunities of analysing the consistent data coming from different sources.

A DW organizes data according to analysis subjects (facts) associated with analysis axes (dimensions). Each fact is composed of indicators (measures). Finally, each dimension may contain one or several analysis viewpoints (hierarchies). Hierarchies allow users to aggregate the attributes of a dimension at different levels to facilitate analysis. Hierarchies are identified by attributes called parameters.

Merging two DWs is a complex task that implies solving several problems. The first issue is identifying the common basic components (attributes, measures) and defining semantic relationships between these components. The second issue is merging schemata that bear common components. Merging two multidimensional DWs is difficult because two dimensions can (1) be completely identical in terms of schema, but not necessarily in terms of instances; (2) have common hierarchies or have sub-parts of hierarchies in common without necessarily sharing common instances. Likewise, two schemata can deal with the same fact or different facts, and even if they deal with the same facts, they may or may not have measures in common, without necessarily sharing common data.

Moreover, a merged DW should respect the constraints of the input multidimensional elements, especially the hierarchical relationships between attributes. When we merge two dimensions having matched attributes of two DWs, the final DW should preserve all the partial orders of the input hierarchies (i.e., the binary aggregation relationships between parameters) of the two dimensions. It is also necessary to integrate all the instances of the input DWs, which may cause the generation of empty values in the merged DW. Thus, the merging process should also include a proper analysis of empty values.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Association for Computing Machinery.

ACM ISBN 978-1-4503-8991-4/21/07...\$15.00

<https://doi.org/10.1145/3472163.3472268>

In sum, the DW merging process concerns matching and merging tasks. The matching task consists in generating correspondences between similar schema elements (dimension attributes and fact measures) [4] to link two DWs. The merging task is more complex and must be carried out at two levels: the schema level and the instance level. Schema merging is the process of integrating several schemata into a common, unified schema [12]. Thus, DW schema merging aims at generating a merged unified multidimensional schema. The instance level merging deals with the integration and management of the instances. In the remainder of this paper, the term “matching” designates schema matching without considering instances, while the term “merging” refers to the complete merging of schemata and corresponding instances.

To address these issues, we define an automatic approach to merge two DWs modeled as star schemata (i.e., schemata containing only one fact table), which (1) generates an integrated DW conforming to the multidimensional structures of the input DWs, (2) integrates the input DW instances into the integrated DW and copes with empty values generated during the merging process.

The remainder of this paper is organized as follows. In Section 2, we review the related work about matching and merging DWs. In Section 3, we specify an automatic approach to merge different DWs and provide DW merging algorithms at the schema and instance levels. In Section 4, we experimentally validate our approach. Finally, in Section 5, we conclude this paper and discuss future research.

2 RELATED WORK

DW merging actually concerns the matching and the merging of multidimensional elements. We classify the existing approaches into four levels: matching multidimensional components, matching multidimensional schemata, merging multidimensional schemata and merging DWs.

A multidimensional component matching approach for matching aggregation levels is based on the fact that the cardinality ratio of two aggregation levels from the same hierarchy is nearly always the same, no matter the dimension they belong to [3]. Thus, by creating and manipulating the cardinality matrix for different dimensions, it is possible to discover the matched attributes.

The matching of multidimensional schemata directs at discovering the matching of every multidimensional components between two multidimensional schemata. A process to automatically match two multidimensional schemata is achieved by evaluating the semantic similarity of multidimensional component names [2]. Attribute and measure data types are also compared in this way. The selection metric of bipartite graph helps determine the mapping choice and define rules aiming at preserving the partial orders of the hierarchies at mapping time. Another approach matches a set of star schemata generated from both business requirements and data sources [5]. Semantic similarity helps find the matched facts and dimension names. Yet, the DW designer must intervene to manually identify some elements.

A two-phase approach for automatic multidimensional schema merging is achieved by transforming the multidimensional schema into a UML class diagram [7]. Then, class names are compared and the number of common attributes relative to the minimal number

of attributes of the two classes is computed to decide whether two classes can be merged.

DW merging must operate at both schema and instance levels. Two DW merging approaches are the intersection and union of the matched dimensions. Instance merging is realized by a d-chase procedure [15]. The second merging strategy exploits similar dimensions based on the equivalent levels in schema merging [11]. It also uses the d-chase algorithm for instance merging. However, the two approaches above do not consider the fact table. Another DW merging approach is based on the lexical similarity of schema string names and instances, and by considering schema data types and constraints [8]. Having the mapping correspondences, the merging algorithm takes the preservation requirements of the multidimensional elements into account, and is formulated to build the final consolidated DW. However, merging details are not precise enough and hierarchies are not considered.

To summarize, none of the existing merging methods can satisfy our DW merging requirements. Some multidimensional components are ignored in these approaches, and the merging details of each specific multidimensional components is not explicit enough, which motivates us to propose a complete DW merging approach.

3 PRELIMINARIES

We introduce in this section the basic concepts of multidimensional DW design [13]. The multidimensional DW can be modelled by a star or a constellation schema. In the star schema, there is a single fact connected with different dimensions, while the constellation schema consists of more than one fact which share one or several common dimensions.

Definition 3.1. A constellation denoted C is defined as $(N^C, F^C, D^C, Star^C)$ where N^C is a constellation name, $F^C = \{F_1^C, \dots, F_m^C\}$ is a set of facts, $D^C = \{D_1^C, \dots, D_n^C\}$ is a set of dimensions, $Star^C : F^C \rightarrow 2^{D^C}$ associates each fact to its linked dimensions. A star is a constellation where F^C contains a single fact; i.e. $m = 1$.

A dimension models an analysis axis and is composed of attributes (dimension properties).

Definition 3.2. A dimension, denoted $D \in D^C$ is defined as (N^D, A^D, H^D, I^D) where N^D is a dimension name, $A^D = \{a_1^D, \dots, a_u^D\} \cup \{id^D\}$ is a set of attributes, where id^D represents the dimension identifier, which is also the parameter of the lowest level and called the root parameter. $H^D = \{H_1^D, \dots, H_v^D\}$ is a set of hierarchies, $I^D = \{i_1^D, \dots, i_p^D\}$ is a set of dimension instances. The value of the instance i_p^D for an attribute a_u^D is annotated as $i_p^D.a_u^D$.

Dimension attributes (also called parameters) are organised according to one or more hierarchies. Hierarchies represent a particular vision (perspective) and each parameter represents one data granularity according to which measures could be analysed.

Definition 3.3. A hierarchy of a dimension D , denoted $H \in H^D$ is defined as $(N^H, Param^H)$ where N^H is a hierarchy name, $Param^H = \langle id^D, p_2^H, \dots, p_v^H \rangle$ is an ordered set of dimension attributes, called parameters, which represent useful graduations along the dimensions, $\forall k \in [1..v], p_k^H \in A^D$. The roll up relationship between two parameters can be denoted by $p_1^H \leq_H p_2^H$

for the case where p_1^H roll up to p_2^H in H . For $Param^H$, we have $id^D \leq_H p_1^H, p_1^H \leq_H p_2^H, \dots, p_{v-1}^H \leq_H p_v^H$. The matching of multidimensional schemata is based on the matching of parameters, the matching relationship between two parameters of two hierarchies $p_i^{H_1}$ and $p_j^{H_2}$ is denoted as $p_i^{H_1} \simeq p_j^{H_2}$.

A sub-hierarchy is a continuous sub-part of a hierarchy which we call the parent hierarchy of the sub-hierarchy. This concept will be used in our algorithms, but it is not really meaningful. So a sub-hierarchy has the same elements than a hierarchy, but its lowest level is not considered as "id". All parameters of a sub-hierarchy are contained in its parent hierarchy and have the same partial orders than those in the parent hierarchy. "Continuous" means that in the parameter set of the parent hierarchy of a sub-hierarchy, between the lowest and highest level parameters of the sub-hierarchy, there is no parameter which is in the parent hierarchy but not in the sub-hierarchy.

Definition 3.4. A sub-hierarchy SH of $H \in H^D$ is defined as $(N^{SH}, Param^{SH})$ where N^{SH} is a sub-hierarchy name, $Param^{SH} = \langle p_1^{SH}, \dots, p_v^{SH} \rangle$ is an ordered set of parameters, called parameters, $\forall k \in [1..v], p_k^H \in Param^H$. According to the relationship between a sub-hierarchy and its parent hierarchy, we have: (1) $\forall p_1^{SH}, p_2^{SH} \in Param^{SH}, p_1^{SH} \leq_{SH} p_2^{SH} \Rightarrow p_1^{SH}, p_2^{SH} \in Param^H \wedge p_1^{SH} \leq_H p_2^{SH}$, (2) $\forall p_1^H, p_2^H, p_3^H \in Param^H, p_1^H \leq_H p_2^H \wedge p_2^H \leq_H p_3^H \wedge p_1^H, p_3^H \in Param^{SH} \Rightarrow p_2^H \in Param^{SH}$.

A fact reflects information that has to be analysed according to dimensions and is modelled through one or several indicators called measures.

Definition 3.5. A fact, noted $F \in F^C$ is defined as $(N^F, M^F, I^F, IStar^F)$ where N^F is a fact name, $M^F = \{m_1^F, \dots, m_w^F\}$ is a set of measures. $I^F = \{i_1^F, \dots, i_q^F\}$ is a set of fact instances. The value of a measure m_w^F of the instance i_q^F is denoted as $i_q^F.m_w^F$. $IStar^F : I^F \rightarrow \mathcal{D}^F$ is a function where \mathcal{D}^F is the cartesian product over sets of dimension instances, which is defined as $\mathcal{D}^F = \prod_{D_k \in Star^C(F)} I^{D_k}$. $IStar^F$ associates fact instances to their linked dimension instances.

We complete these definitions by a function $extend(H_1, H_2)$ allowing to extend the parameters of the first (sub)hierarchy H_1 by the other one (H_2).

4 AN AUTOMATIC APPROACH FOR DW MERGING

Like illustrated in Figure 1, merging two DWs implies matching steps and steps dedicated to the merging of dimensions and facts. The matching of parameters and measures are based on syntactic and semantic similarities [10][6] for the attribute or measure names. Since the matching is intensively studied in the literature, we focus in this paper only on the merging steps of our process (green rectangle in Figure 1). In regard to the merging, we firstly define an algorithm for the merging of hierarchies by decomposing two hierarchies into sub-hierarchy pairs and merging them to get the final hierarchy set. Then, we define an algorithm of dimension merging concerning both instance and schema levels and which completes some empty values. Finally, we define an algorithm of

the star merging based on the dimension merging algorithm which merges the dimensions and the facts at the schema and instance levels and corrects the hierarchies after the merging.

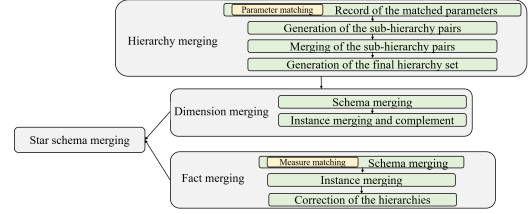


Figure 1: Overview of the merging process

4.1 Hierarchy merging

In this section, we define the schema merging process of two hierarchies coming from two different dimensions. The first challenge is that we should preserve the partial orders of the parameters. The second one is how to decide the partial orders of the parameters coming from different original hierarchies. These challenges are solved in the algorithm proposed below which is achieved by 4 steps: record of the matched parameters, generation of the sub-hierarchy pairs, merging of the sub-hierarchy pairs and generation of the final hierarchy set.

Algorithm 1 *MergeHierarchies*(H_1, H_2)

Output: A set of merged hierarchies H' or two sets of merged hierarchies $H^{1'}$ and $H^{2'}$

```

1:  $M, SH', H' \leftarrow \emptyset$ ; //  $M$  is an ordered set of the couples of matched
   parameters with possibly the couple of the last parameters, for the  $n$ th
   parameter couple  $M[n-1]$ ,  $M[n-1][0]$  represents the parameter of
    $H_1$  in  $M[n-1]$ , while  $M[n-1][1]$  represents the one of  $H_2$ .
2:  $Param^{SH_1}, Param^{SH_{1'}}, Param^{SH_2}, Param^{SH_{2'}} \leftarrow \emptyset$ ;
3: for each  $p_i^{H_1} \in Param^{H_1}$  do
4:   for each  $p_j^{H_2} \in Param^{H_2}$  do
5:     if  $p_i^{H_1} \simeq p_j^{H_2}$  then
6:        $M \leftarrow M + \langle p_i^{H_1}, p_j^{H_2} \rangle$ ;
7:     end if
8:   end for
9: end for
10: if  $M = \emptyset$  then
11:    $H^{1'} \leftarrow \{H_1\}; H^{2'} \leftarrow \{H_2\}$ ;
12:   return  $H^{1'}, H^{2'}$ 
13: else
14:    $m_l \leftarrow \langle Param^{H_1} [|Param^{H_1}| - 1], Param^{H_2} [|Param^{H_2}| - 1] \rangle$ ;
   // pair of the last parameters
15:   if  $m_l \notin M$  then
16:      $M \leftarrow M + m_l$ ;
17:   end if
18:   for  $i = 0$  to  $|M| - 2$  do
19:      $p_1^{SH_1} \leftarrow M[i][0]$ ; // first parameter of  $SH_1$ 
20:      $p_{v_1}^{SH_1} \leftarrow M[i+1][0]$ ; // last parameter of  $SH_1$ 
21:      $p_1^{SH_2} \leftarrow M[i][1]$ ; // first parameter of  $SH_2$ 
22:      $p_{v_2}^{SH_2} \leftarrow M[i+1][1]$ ; // last parameter of  $SH_2$ 
23:     if  $Param^{SH_1} \subseteq Param^{SH_2}$  then
24:        $SH' \leftarrow \{SH_2\}$ ;
25:     else if  $Param^{SH_2} \subseteq Param^{SH_1}$  then

```

```

26:    $SH' \leftarrow \{SH_1\};$ 
27:   else if  $FD_{SH_1,SH_2} \neq \emptyset$  then
28:     for each  $Param' \in MergeParameters(FD_{SH_1,SH_2})$  do
29:        $Param^{SH_a} \leftarrow Param'; SH' \leftarrow SH' + SH_a;$ 
30:     end for
31:   else
32:      $SH' \leftarrow \{SH_1, SH_2\};$ 
33:   end if
34:    $H' \leftarrow \{H'_a.extend(SH'_b) | (H'_a \in H') \wedge (SH'_b \in SH')\};$ 
35:   end for
36: end if
37: if  $id^{D_1} \approx id^{D_2}$  then
38:    $H' \leftarrow H' \cup \{H_1, H_2\};$ 
39:   return  $H'$ 
40: else
41:    $p_1^{SH_1'} \leftarrow p_1^{H_1};$  //first parameter of  $SH_1'$ 
42:    $p_{v_1'}^{SH_1'} \leftarrow M[0][0];$  //last parameter of  $SH_1'$ 
43:    $p_1^{SH_2'} \leftarrow p_1^{H_2};$  //first parameter of  $SH_2'$ 
44:    $p_{v_2'}^{SH_2'} \leftarrow M[0][1];$  //last parameter of  $SH_2'$ 
45:   for each  $H'_c \in H'$  do
46:      $H^{1'} \leftarrow SH_1'.extend(H'_c); H^{2'} \leftarrow SH_2'.extend(H'_c);$ 
47:   end for
48:    $H^{1'} \leftarrow H^{1'} \cup \{H_1\}; H^{1'} \leftarrow H^{2'} \cup \{H_2\};$ 
49:   return  $H^{1'}, H^{2'}$ 
50: end if

```

4.1.1 Record of the matched parameters. The first step of the algorithm consists in matching the parameters of the two hierarchies and record the matched parameter pairs (L_1 - L_9). If there is no matched parameter between the two hierarchies, the merging process stops (L_{11} - L_{12}).

4.1.2 Generation of the sub-hierarchy pairs. Then the algorithm generates pairs containing 2 sub-hierarchies (SH_1 and SH_2) of the original hierarchies whose lowest and highest level parameters are adjacent in the list of matched parameter pairs that we created in the previous step (L_{18} - L_{22}). To make sure that the last parameters of the two hierarchies are included in the sub-hierarchies, we also add the pair of the last parameters into the matched parameter pair (L_{14} - L_{17}).

Example 4.1. In Figure 2, for (a), we have $H_1.Code \approx H_2.Code$, $H_1.Department \approx H_2.Department$, $H_1.Continent \approx H_2.Continent$. So for the first sub-hierarchy pair, the first parameter of SH_1 and SH_2 is *Code* and their last parameter is *Department*, so we have: $Param^{SH_1} = \langle Code, Department \rangle$, $Param^{SH_2} = \langle Code, City, Department \rangle$. In the second sub-hierarchy pair, we get the sub-hierarchy of H_1 from *Department* to *Continent*: $Param^{SH_1} = \langle Department, Region, Continent \rangle$, and the sub-hierarchy of H_2 from *Department* to *Continent*: $Param^{SH_2} = \langle Department, Country, Continent \rangle$. If the last parameters of the two original hierarchies do not match, like *Continent* of H_1 and *Country* of H_3 in (b), $\langle Continent, Country \rangle$ is added into the matched parameter pair M of the algorithm so that the last sub-hierarchies of H_1 and H_3 are $Param^{SH_1} = \langle Department, Region, Continent \rangle$ and $Param^{SH_3} = \langle Department, Country \rangle$.

4.1.3 Merging of the sub-hierarchies. We then merge each sub-hierarchy pair to get a set of merged sub-hierarchies (SH') and

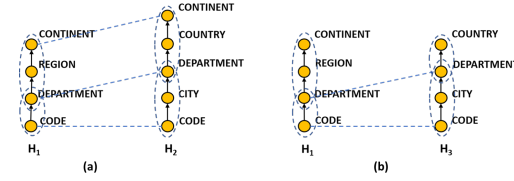


Figure 2: Example of generation of the sub-hierarchy pairs

combine each of these sub-hierarchy sets to get a set of merged hierarchies (H') (L_{23} - L_{35}).

The matched parameters will be merged into one parameter, so it's the unmatched parameters that we should deal with. We have 2 cases in terms of the unmatched parameters.

If one of the sub-hierarchies has no unmatched parameter, we obtain a sub-hierarchy set containing one sub-hierarchy whose parameter set is the same as the other sub-hierarchy (L_{23} - L_{26}).

Example 4.2. For the first parameter pair $SH_1 = \langle Code, Department \rangle$ and $SH_2 = \langle Code, City, Department \rangle$ of H_1 and H_2 in Figure 4. We see that SH_1 does not have any unmatched parameter, so the obtained sub-hierarchy set contains one sub-hierarchy whose parameter set is the same as SH_2 which is $Param^{SH'} = \langle Code, City, Department \rangle$.

The second case is that both two sub-hierarchies have unmatched parameters (L_{27} - L_{30}). We then see if these unmatched parameters can be merged into one or several hierarchies and discover their partial orders. Our solution is based on the functional dependencies (FDs) of these parameters. To be able to detect the FDs of the parameters of the two sub-hierarchies, we should make sure that there are intersections between the instances of these two sub-hierarchies which means that they should have same values on the root parameter of the sub-hierarchies. We keep only the FDs which have a single parameter in both hands and which can not be inferred by transitivity. These FDs are represented in the form of ordered set (FD_{SH_1,SH_2}) are then treated by algorithm 2 *MergeParameters* to get the parameter sets of the merged sub-hierarchies. If it's not possible to discover the FDs, the two sub-hierarchies are impossible to be merged (L_{31} - L_{32}).

Algorithm 2 *MergeParameters* constructs recursively the parameter sets from the FDs in the form of ordered sets. In each recursion loop, for each one of these sets, we search for the other ones whose non-last (or non-first) elements have the same values and order as its non-first (or non-last) elements and then merge them (L_6 - L_{21}). The recursion is finished until there are no more two sets being able to be merged (L_{22} - L_{31}).

Example 4.3. If we have $FD = \langle \langle A, B \rangle, \langle B, C \rangle, \langle B, F \rangle, \langle C, E \rangle, \langle D, B \rangle \rangle$. Like illustrated in Figure 3, in the first recursion, by merging the ordered set, we get $Param = \langle \langle A, B, C \rangle, \langle A, B, F \rangle, \langle B, C, E \rangle, \langle D, B, C \rangle, \langle D, B, F \rangle \rangle$, all the ordered sets in FD are merged, so there are only merged ordered set in $Param$. $Param$ is then inputted to the second recursion, we then get the next $Param = \langle \langle A, B, C, E \rangle, \langle D, B, C, E \rangle \rangle$ after the merging of the ordered sets, since $\langle A, B, F \rangle$ and $\langle D, B, F \rangle$ are not merged, they are also added into $Param$, and we get $Param = \langle \langle A, B, C, E \rangle, \langle D, B, C, E \rangle, \langle A, B, F \rangle, \langle D, B, F \rangle \rangle$. In the final

recursion, it's no more possible to merge any two ordered sets, so the parameter set of the final result of the hierarchy set is $\langle\langle A, B, C, E \rangle, \langle D, B, C, E \rangle, \langle A, B, F \rangle, \langle D, B, F \rangle\rangle$.

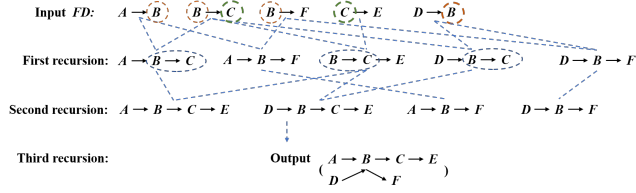


Figure 3: Example of parameter merging based on FDs

Algorithm 2 *MergeParameters(FD)*

Output: A set of parameter sets *Param*

```

1:  $l \leftarrow |FD|$ ;
2: for  $n \leftarrow 0$  to  $l - 1$  do
3:    $fdmerged[n] \leftarrow \text{False}$ ; //Boolean indicating whether an element
   in FD is merged
4: end for
5:  $existmerged \leftarrow \text{False}$ ; //Boolean indicating whether there are
   elements that are merged in a recursion loop
6: for  $i \leftarrow 0$  to  $l - 1$  do
7:   for  $j \leftarrow i + 1$  to  $l$  do
8:     if  $FD[i][1 : l - 1] = FD[j][0 : l - 2]$  // $FD[a][b : c]$ 
       represents the ordered set having the values and order from the
       bth element to the cth element of  $FD[a]$  then
9:        $Param^t \leftarrow FD[i][1 : l - 1]$ ;
10:       $Param^t \leftarrow Param^t + FD[j][l - 1]$ ;
11:       $Param \leftarrow Param + Param^t$ ;
12:       $fdmerged[i], fdmerged[j], existmerged \leftarrow \text{True}$ ;
13:    end if
14:    if  $FD[i][0 : l - 2] = FD[j][1 : l - 1]$  then
15:       $Param^t \leftarrow FD[j][1 : l - 1]$ ;
16:       $Param^t \leftarrow Param^t + FD[i][l - 1]$ ;
17:       $Param \leftarrow Param + Param^t$ ;
18:       $fdmerged[i], fdmerged[j], existmerged \leftarrow \text{True}$ ;
19:    end if
20:  end for
21: end for
22: if  $existmerged = \text{True}$  then
23:   for  $m \leftarrow 0$  to  $l - 1$  do
24:     if  $fdmerged[m] = \text{False}$  then
25:        $Param \leftarrow Param + FD[m]$ ;
26:     end if
27:   end for
28:    $Param \leftarrow \text{MergeParameters}(Param)$ ;
29: else
30:    $Param \leftarrow FD$ ;
31: end if
32: return  $Param$ 

```

After the merging of each sub-hierarchy pair, we extend the final merged hierarchy set by the new merging result (L_{34}).

4.1.4 Generation of the final hierarchy set. L_{37} - L_{49} concerns the generation of the final hierarchy set. The two original hierarchies may have different instances, so there may be empty values in the instances of the merged hierarchies. Some empty values can be completed, which is introduced in the next section of dimension

merging. But not all empty values can be completed. The empty values generate the incomplete hierarchies and make the analysis difficult. Inspired by the concept of the structural repair[1], we also add the two original hierarchies into the final hierarchy set. Then for a parameter which appears in different hierarchies, it can be divided into different parameters in different hierarchies of the hierarchy set so that each hierarchy is complete. Thus, for the multidimensional schema that we get, we provide an analysis form like shown in Figure 4. In the analysis form, one parameter can be marked with different numbers if it is in different hierarchies.

For the generation of the final hierarchy set, we discuss 2 cases where the 2 hierarchies have the matched root parameters which means their dimensions are the same analysis axis and the opposite case which will lead to 2 kinds of output results (one or two sets of merged hierarchies).

If the root parameters of the two original hierarchies match, we simply add the two original hierarchies into the merged hierarchy set obtained in the previous step to get one final merged hierarchy set. (L_{37} - L_{39}).

Example 4.4. For the hierarchies H_1 and H_2 in Figure 4, we combine the merged hierarchy obtained in *Example 4.4* with the result gained in *Example 4.2* to get the merged hierarchy H_m : $\langle \text{Code}, \text{City}, \text{Department}, \text{Region}, \text{Country}, \text{Continent} \rangle$. We add H_m into the hierarchy set H' and then also add the original hierarchies H_1 and H_2 . Thus H' is the final merged hierarchy set.

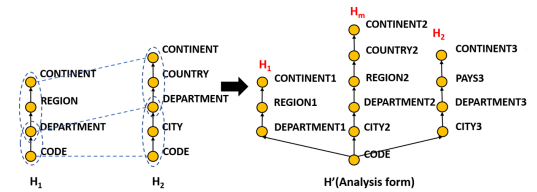


Figure 4: Hierarchy merging example

If the root parameters of the two original hierarchies do not match, we will get two merged hierarchy sets instead of one. For each original hierarchy, the final merged hierarchy set will be the extension of the sub-hierarchy containing all the parameters which are not included in any one of the sub-hierarchies created before ($SH_{1'}$ and $SH_{2'}$) with the merged hierarchy set that we get plus this original hierarchy itself (L_{41} - L_{49}).

Example 4.5. In Figure 5, between H_1 and H_3 , we have $H_1.\text{Department} \simeq H_3.\text{Department}$ and $H_1.\text{Continent} \simeq H_3.\text{Continent}$. We can then get one sub-hierarchy pair in which there are 2 sub-hierarchies containing parameter sets $\langle \text{Department}, \text{Region}, \text{Continent} \rangle$ and $\langle \text{Department}, \text{Country}, \text{Continent} \rangle$. By merging the sub-hierarchy pairs, we get the merged hierarchy whose parameter set is $\langle \text{Department}, \text{Region}, \text{Country}, \text{Continent} \rangle$. For H_1 , the remaining part $\langle \text{Code} \rangle$ is associated to it to get the merged hierarchy H_{13}^1 . We then get the merged hierarchy set of H_1 containing H_1 and H_{13}^1 . We do the same thing for H_3 and get the merged hierarchy set containing H_3 and H_{13}^2 .

4.2 Dimension merging

This section concerns the merging of two dimensions having matched attributes which is realized by algorithm 3 *MergeDimensions*. We consider both the schema and instance levels for the merging of dimensions. The schema merging is based on the merging of hierarchies. Concerning the instances, we have 2 tasks: merging the instances and completing the empty values.

Algorithm 3 *MergeDimensions*(D_1, D_2)

Output: One merged dimension D' or two merged dimensions $D^{1'}$ and $D^{2'}$

```

1: if  $id^{D_1} \approx id^{D_2}$  then
2:    $H^{D'} \leftarrow \emptyset$ ;
3:   for each  $H_i^{D_1} \in H^{D_1}$  do
4:     for each  $H_j^{D_2} \in H^{D_2}$  do
5:        $H^{D'} \leftarrow H^{D'} \cup MergeHierarchies(H_i^{D_1}, H_j^{D_2})$ ;
6:     end for
7:   end for
8:    $A^{D'} \leftarrow A^{D_1} \cup A^{D_2}$ ;  $H^m \leftarrow H^{D'} \setminus (H^{D_1} \cup H^{D_2})$ ;
9:   CompleteEmpty( $D', D', H^m$ );
10:  return  $D'$ 
11: else
12:   $H^{D^{1'}}, H^{D^{2'}}, A^{D^{1'}}, A^{D^{2'}} \leftarrow \emptyset$ ;
13:  for each  $H_i^{D_1} \in H^{D_1}$  do
14:    for each  $H_j^{D_2} \in H^{D_2}$  do
15:       $H^{1'}, H^{2'} \leftarrow MergeHierarchies(H_i^{D_1}, H_j^{D_2})$ ;
16:       $H^{D^{1'}} \leftarrow H^{D^{1'}} \cup H^{1'}$ ;  $H^{D^{2'}} \leftarrow H^{D^{2'}} \cup H^{2'}$ ;
17:    end for
18:  end for
19:  for each  $H_u^{D^{1'}} \in H^{D^{1'}}$  do
20:     $A^{D^{1'}} \leftarrow A^{D^{1'}} \cup Param^{H_u^{D^{1'}}}$ ;
21:  end for
22:  for each  $H_v^{D^{2'}} \in H^{D^{2'}}$  do
23:     $A^{D^{2'}} \leftarrow A^{D^{2'}} \cup Param^{H_v^{D^{2'}}}$ ;
24:  end for
25:   $H^m \leftarrow H^{D'} \setminus H^{D_1}$ ;  $H^m \leftarrow H^{D'} \setminus H^{D_2}$ ;
26:  CompleteEmpty( $D^{1'}, D^{2'}, H^m$ );
27:  CompleteEmpty( $D^{2'}, D^{1'}, H^m$ );
28:  return  $D^{1'}, D^{2'}$ 
29: end if

```

4.2.1 Schema merging. If the root parameters of the two dimensions match, the algorithm generates a merged dimension (L_1 - L_8). The hierarchy set of the merged dimension is the union of the hierarchy sets generated by merging every 2 hierarchies of the original dimensions (L_3 - L_7). We also get a hierarchy set containing only the merged hierarchies but no original hierarchies (H^m) which is to be used for the complement of the empty values (L_8). The attribute set of the merged dimension is the union of the attribute sets of the original dimensions (L_8).

Example 4.6. Given 2 original dimensions D_1 and D_2 in Figure 8 and their instances in Figure 6, we can get the merged dimension schema D' in Figure 8. In D' , H_1 and H_2 are the original hierarchies of D_1 , H_3 and H_4 are those of D_2 , H_{13} is a merged hierarchy of H_1 and H_3 , and H_{24} is a merged hierarchy of H_2 and H_4 . We can thus get $H^{D'} = \{H_1, H_2, H_3, H_4, H_{13}, H_{24}\}$, $H^m = \{H_{13}, H_{24}\}$, $A^{D'} =$

$\{Code, City, Department, Region, Country, Continent, Profession, Subcategory, Category\}$

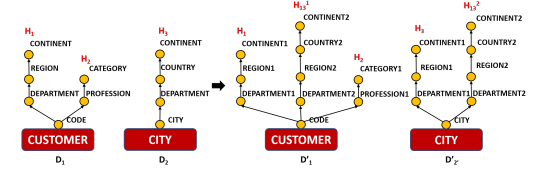


Figure 5: Dimension merging example (schema)

When the root parameters of the two dimensions don't match, we will get a merged dimension for each original dimension, which is realized by L_{13} - L_{25} . For each original dimension, the hierarchy set of its corresponding merged dimension is the union of all hierarchy sets generated by merging every 2 hierarchies of the original dimensions (L_{13} - L_{18}), the attribute set is the union of the attributes of each hierarchy in the merged dimension (L_{19} - L_{24}). Similar to the first case, we get a hierarchy set containing only the merged hierarchies for each original dimension (H^{m1} and H^{m2}) (L_{26} - L_{27}).

Example 4.7. Given 2 original dimensions D_1 and D_2 in Figure 5 and their instances in Figure 7, after the execution of algorithm 3 *MergeDimensions*, we can get the merged dimension schema $D^{1'}$ and $D^{2'}$ in Figure 5. In $D^{1'}$, H_1 and H_2 are the original hierarchies of D_1 , H_{13}^1 is the merged hierarchy of H_1 and H_3 . In $D^{2'}$, H_3 is the original hierarchy of D_2 , H_{13}^2 is the merged hierarchy of H_1 and H_3 . So for D_1 , we have $H^{D^{1'}} = \{H_1, H_2, H_{13}^1\}$, $H^{m1} = \{H_{13}^1\}$, $A^{D^{1'}} = \{Code, Department, Region, Country, Continent, Profession, Category\}$, while for D_2 , we get $H^{D^{2'}} = \{H_3, H_{13}^2\}$, $H^{m2} = \{H_{13}^2\}$, $A^{D^{2'}} = \{City, Department, Region, Country, Continent\}$

4.2.2 Instance merging and complement. When the root parameters of the two dimensions match, the instance of the merged dimension is obtained by the union of the two original dimension instances which means that we insert the data of the two original dimension tables into the merged dimension table and merge the lines which have the same root parameter instance (L_9).

Example 4.8. The instance merging result of Example 4.2.1 is presented in Figure 6. All the data in the original dimension tables D_1, D_2 are integrated into the merged dimension table D' . The original tables of the instances are marked on the left of the merged table D' with different colors. There are instances coming from both D_1 and D_2 , which means that they have the same root parameter in D_1 and D_2 , and are therefore merged together.

The attribute set of the merged dimension contains all the attributes of two original dimensions, while the original dimensions may contain their unique attributes. So there may be empty values in the merged dimension table on the instances coming from only one of the original dimension tables and we should complete the empty values on the basis of the existing data (L_9).

The complement of the empty values is realized by Algorithm *CompleteEmpty* where the input $D^{1'}$ is the merged dimension table having empty values to be completed, $D^{2'}$ is the merged dimension table which provides the completed values and H^m is the hierarchy

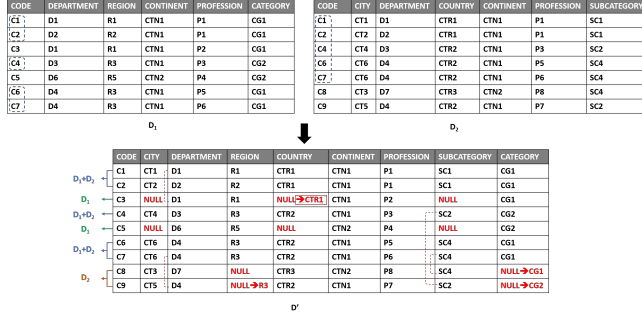


Figure 6: Dimension merging example (instance)

set of $D^{1'}$ containing only merged hierarchies but no original hierarchies. In this discussed case, D' is inputted as both $D^{1'}$ and $D^{2'}$ in *CompleteEmpty* since we get one merged dimension including all data of two original dimensions (L_{11}).

Algorithm 4 *CompleteEmpty*($D^{1'}$, $D^{2'}$, H^m)

```

1: for each  $H_a^m \in H^m$  do
2:    $I^n \leftarrow \emptyset$ ;
3:    $I^n \leftarrow I^n \cup \{i_k^{D^{1'}} \in I^{D^{1'}} \mid (i_k^{D^{1'}} \cdot p_1^{H_a^m} \text{ is not null}) \wedge (\exists p_v^{H_a^m} \in \text{Param}^{H_a^m}, i_k^{D^{1'}} \cdot p_v^{H_a^m} \text{ is null})\}$ ;
4:   for each  $i_b^n \in I^n$  do
5:      $P^n \leftarrow \{p_v^{H_a^m} \in \text{Param}^{H_a^m} \mid i_b^n \cdot p_v^{H_a^m} \text{ is null}\}$ ;
6:      $P^r \leftarrow \{p_v^{H_a^m} \in \text{Param}^{H_a^m} \mid (i_b^n \cdot p_v^{H_a^m} \text{ is not null}) \wedge (\forall p_s^n \in P^n, p_v^{H_a^m} \leq_H p_s^n)\}$ ;
7:     if  $\exists i_u^{D^{2'}} \in I^{D^{2'}} \exists p_w^r \in P^r, (i_u^{D^{2'}} \cdot p_w^r = i_b^n \cdot p_w^r) \wedge (\forall p_q^n \in P^n, i_u^{D^{2'}} \cdot p_q^n \text{ is not null})$  then
8:       for each  $p_c^n \in P^n$  do
9:          $i_b^n \cdot p_c^n \leftarrow i_u^{D^{2'}} \cdot p_c^n$ ;
10:      end for
11:    end if
12:  end for
13: end for

```

For an empty value, we search for an instance which has the same value as the instance of this empty value on one of the parameters rolling up to the parameter of the empty value and whose value of the parameter of the empty value is not empty, we can then fill the empty by this non-empty value. The complement of the empty values is also possibly a change of hierarchies. Nevertheless, after completing the empty values of an instance, there may be some completed parameters which are not included in the hierarchies of the instance, so the complement of such values does not make sense in this case. The possible change of the hierarchy is from the hierarchies containing less parameters to those containing more parameters. We know that the merged hierarchies contain more parameters than their corresponding original hierarchies. Hence, before the complement of an instance, we will first look at the merged hierarchies to decide which parameter values can be completed.

In algorithm 4 *CompleteEmpty* which aims to complete the empty values, for each hierarchy in the merged hierarchy set we see, if (a) there exists instances in the merged dimension table which contains empty values on the parameters of this hierarchy

(L_3) and (b) where the value of the second lowest parameter is not empty (L_3). The condition a is basic because we need empty values to be completed. Since we will complete the empty values by the other lines of the merged dimension table, we can only complete the empty values based on the non-id parameters since the id is unique, so if the second lowest parameter is empty, it can never be completed so that the hierarchy can never be completed. That's why we have the condition b. For each one of the instances satisfying these conditions (I^n), we search for the parameters (P^n) having empty values (L_5) and to make sure that each one of them can be completed, we search also for the parameters (P^r) which roll up to the lowest of them and to which we refer to complete the empty values (L_6). We can then complete the empty values like discussed in the previous paragraph (L_7 - L_{11}).

Example 4.9. After the merging in *Example 4.9*, we get the empty values of D' which are in red in Figure 6. The merged hierarchies are H_{13} and H_{24} as illustrated in Figure 5. For H_{13} , the instances of code C3 and C5 have empty values on the second root parameter *City*, which do not satisfy the condition b. As we can see, for the instance of C3, although the value of *Country* can be retrieved through the value of *Department* which is the same as the instance of C1, the value of *City* can not be completed and thus we should give up this complement. For the instance of C9, the value of *Region* is completed by C7 which has the same value of *Department* and whose value of *Region* is not empty. When it's the turn of H_{24} , values of *Category* of C8 and C9 are completed in the same way.

When the root parameters of the two dimensions don't match, the instance merging and complement are done by L_{26} - L_{27} . The values of the attributes of one of the dimension tables coming from the other dimension table are empty, so there is only instance complement but no merging. We also call algorithm 4 *CompleteEmpty* to complete the instances for each one of the merged dimension tables.

Example 4.10. The instance merging and complement of the example for *Example 4.8* is demonstrated in Figure 7. For $D^{1'}$, *Country* comes from the dimension table D_2 , so the values of *Country* are completed by the values in $D^{2'}$. The same operation is also done for *Region* of $D^{2'}$.

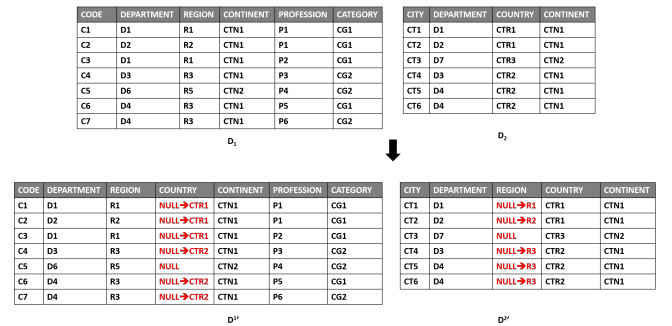


Figure 7: Dimension merging example (instance)

4.3 Star merging

In this section, we discuss the merging of two stars. Having two stars, we can get a star schema or a constellation schema because the fact table of each schema may be merged into one schema or not. The star merging is related to the dimension merging and fact merging. Two stars are possible to be merged only if there are dimensions having matched root parameters between them.

Algorithm 5 *MergeAllDimensions*(S_1, S_2)

Output: A set of merged dimensions $D^{S'}$

```

1: for each  $D_i^{S_1} \in D^{S_1}$  do
2:   for each  $D_j^{S_2} \in D^{S_2}$  do
3:     if  $id^{D_i^{S_1}} \neq id^{D_j^{S_2}}$  then
4:        $D_i^{S_1}, D_j^{S_2} \leftarrow \text{MergeDimensions}(D_i^{S_1}, D_j^{S_2});$ 
5:     end if
6:   end for
7: end for
8:  $D^{S'} \leftarrow \emptyset;$ 
9: for each  $D_u^{S_1} \in D^{S_1}$  do
10:  for each  $D_v^{S_2} \in D^{S_2}$  do
11:    if  $id^{D_u^{S_1}} \approx id^{D_v^{S_2}}$  then
12:       $D^{S'} \leftarrow D^{S'} \cup \text{MergeDimensions}(D_u^{S_1}, D_v^{S_2});$ 
13:    end if
14:  end for
15: end for
16: for each  $D_k^{S'} \in D^{S'}$  do
17:  for each  $H_m^{D_k^{S'}} \in H^{D_k^{S'}}$  do
18:    if  $\nexists i_r^{D_k^{S'}} \in I^{D_k^{S'}} \wedge (i_r^{D_k^{S'}} \text{ is on } H_m^{D_k^{S'}}) \vee (i_r^{D_k^{S'}} \text{ is only on } H_m^{D_k^{S'}} \wedge (H_m^{D_k^{S'}} \in H^{D^{S_1}} \vee H_m^{D_k^{S'}} \in H^{D^{S_2}}))$  then
19:       $H_m^{D_k^{S'}} \leftarrow H_m^{D_k^{S'}} - i_r^{D_k^{S'}};$ 
20:    end if
21:  end for
22: end for
23: return  $D^{S'}$ 

```

For the dimensions of the two stars, we have two cases: 1. The two stars have the same number of dimensions and for each dimension of one schema, there is a dimension having matched root parameters in the other schema. 2. There exists at least one dimension between the two stars which does not have a dimension having a matched root parameter in the other.

The dimension merging of two stars is common for the two cases which is done by algorithm 5 *MergeAllDimension*. We first merge every two dimensions of the two stars which have unmatched root parameters because the merging of such dimensions is able to complete the original dimensions with complementary attributes (L_1 - L_7). Then the dimensions having matched root parameters are merged to generate the merged dimensions of the merged multidimensional schema (L_8 - L_{15}). After the merging and complement of the instances of the dimension tables, there may be some merged hierarchies to which none of the instances belong. In this case, if there will be no more update of the data, such hierarchies should be deleted. There may also be original hierarchies in the merged dimensions such that there is no instance which belongs to them but does not belong to any merged hierarchy containing all the parameters of this original hierarchy. The instances belonging to this

kind of hierarchies belong also to other hierarchies which contains more parameters, so they become useless and should also be deleted (L_{18} - L_{19}).

Example 4.11. For the merging of the dimensions of two stars S_1 and S_2 in Figure 8. The dimension *Product* of S_1 and the dimension *Customer* of S_2 are firstly merged since their root parameters don't match but they have other matched parameters. There are then attributes of dimension *Customer* of S_2 added into dimension *Product* of S_1 . The two dimensions *Customer* and the two dimensions *product* have matched root parameters, so they are merged into the final star schema. After the merging and complement of the instance, we verify each hierarchy in the merged dimension tables. If the merging of S_1 .*Customer* and S_2 .*Customer* is as shown in Figure 5 at the schema level and in Figure 6 at the instance level. In their merged dimension table D' . We can find that all the instances belonging to H_4 also belong to H_{24} which is a merged hierarchy containing all the parameters of H_4 , so H_4 should be deleted.

We then discuss the merging of the other elements in the two cases which is processed by algorithm 6 *MergeStar*:

Algorithm 6 *MergeStar*(S_1, S_2)

Output: A merged multidimensional schema which may be a star schema S' or a merged constellation schema C'

```

1: if ( $|D^{S_1}| = |D^{S_2}|$ )  $\wedge$  ( $\forall D_i^{S_1} \in D^{S_1} \exists D_j^{S_2} \in D^{S_2}, id^{D_i^{S_1}} \approx id^{D_j^{S_2}}$ ) then
2:    $D^{S'} \leftarrow \text{MergeAllDimensions}(S_1, S_2);$ 
3:    $M^{F^{S'}} \leftarrow M^{F^{S_1}} \cup M^{F^{S_2}}; I^{F^{S'}} \leftarrow I^{F^{S_1}} \cup I^{F^{S_2}};$ 
4:    $I\text{Star}^{F^{S'}} \leftarrow I\text{Star}^{F^{S_1}} \cup I\text{Star}^{F^{S_2}};$ 
5:   return  $S'$ 
6: else
7:    $D^{S'} \leftarrow \text{MergeAllDimensions}(S_1, S_2); F^{C'} \leftarrow \{F^{S_1}, F^{S_2}\};$ 
8:   return  $C'$ 
9: end if

```

For the first case, we merge the two fact tables into one fact table and get a star schema. The measure set of the merged star schema is the union of the 2 original measures (L_3). The fact instances are the union of the measure instances of the two input star schemata (L_4). The function associating fact instances to their linked dimension instances of the merged schema is also the union of the functions of the original schemata (L_4).

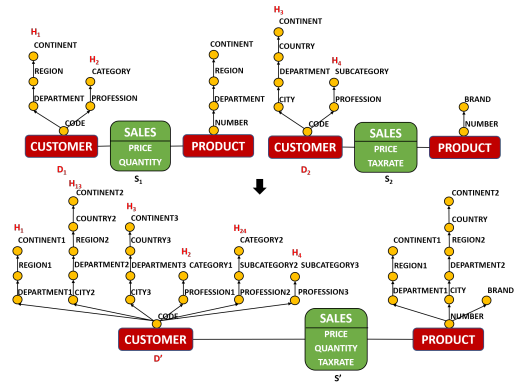


Figure 8: Star merging example (schema)

Figure 9: Star merging example (instance)

Example 4.12. For the two original star schemata in Figure 8, the dimension merging is discussed above so we mainly focus on the merging of fact table instances here. The dimensions *Customer*, *Product* of S_1 have respectively matched root parameters in the dimensions *Customer*, *Product* of S_2 . They also have the same number of dimensions. Therefore we get a merged star schema S' , the original fact tables are merged by merging the measures of S_1 and S_2 to get the fact table of S' . At the instance level, in Figure 9, we have the instances of the fact tables, for the instances of F^{S_1} and F^{S_2} , the framed parts are the instances having the common linked dimension instances, so they are merged into the merged fact table $F^{S'}$, the other instances are also integrated in $F^{S'}$ but with empty values in the merged instances, but they will not have big impacts on the analysis, so they will not be treated particularly.

For the second case, since there are unmatched dimensions, the merged schema should be a constellation schema. The facts of the original schemata have no change at both the schema and instance levels and compose the final constellation. (L_8)

Example 4.13. This example is simplified in Figure 10 due to the space limit. For the original star schemata S_1 and S_2 , they have dimensions *Customer* which have the matched root parameters. They also have their unique dimensions: *Time* of S_1 and *Product* of S_2 . So the merged schema is a constellation schema generated by merging the dimensions *Customer* and by keeping the other dimensions and fact tables. At the instance level, we just have a new merged dimension table of *Customer*, the other dimension and fact tables remain unchanged.

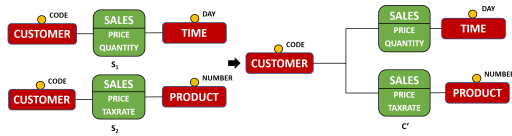


Figure 10: Star merging example (schema)

5 EXPERIMENTAL ASSESSMENTS

To validate the effectiveness of our approach, we applied our algorithms on benchmark data. Unfortunately, we did not find a suitable benchmark for our problem. So, we adapted the datasets of the TPC-H benchmark to generate different DWs. Originally, the TPC-H benchmark serves for benchmarking decision support systems by examining the execution of queries on large volumes of data. Because of space limit, we put the test results in github¹.

¹<https://github.com/Implementation111/Multidimensional-DW-merging>

5.1 Technical environment and Datasets

The algorithms were implemented by Python 3.7 and were executed on a processor of Intel(R) Core(TM) i5-8265U CPU@ 1.60GHz with a 16G RAM. The data are implemented in R-OLAP format through the Oracle 11g DBMS. The TPC-H benchmark provides a pre-defined relational schema² with 8 tables and a generator of massive data.

First, we generated 100M of data files, there are respectively 600572, 15000, 25, 150000, 20000, 80000, 5, 1000 tuples in the table of *Lineitem*, *Customer*, *Nation*, *Orders*, *Part*, *Partsupp*, *Region* and *Supplier*. Second, to have more deeper hierarchies, we included the data of *Nation* and *Region* into *Customer* and *Supplier*, and those of *Partsupp* into *Part*. Third, we transformed these files to generate two use cases by creating 2 DWs for each case. To make sure that there are both common and different instances in different DWs, for each dimension, instead of selecting all the corresponding data, we selected randomly 3/4 of them. For the fact table, we selected the measures related to these dimension data. Since the methods in the related work do not have exactly the same treated components or objective as the ours, we do not have comparable baseline in our experiments.

5.2 Star schema generation

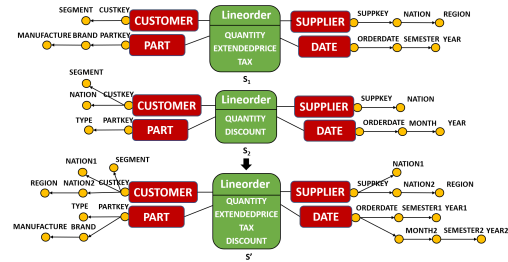


Figure 11: Star schema generation

The objective of this experiment is to merge two star schemata having the same 4 dimensions with the matched lowest level of granularity for each dimension.

After executing our algorithms, we obtain one star schema as shown in Figure 11 which is consistent with the expectations. The parameters of the hierarchies satisfy the relationships of functional dependency. The run time is 30.70s. The 3 dimensions *Supplier*, *Part*, *Date* of the original DWs are merged. Between the different dimensions S_1 .*Supplier* and S_2 .*Customer*, there is a matched attribute *Nation*, so they are also merged such that S_1 .*Supplier* provides S_2 .*Customer* with the attribute *Region*. Then the *Customer* in the merged DW also has the attribute *Region*. We can also observe that normally, in the merged schema, there should be the original hierarchy *Orderdate* → *Month* → *Year* of S_2 .*Date* but which is deleted. By looking up in the table, we find that there is no tuple which belongs to this hierarchy but not to *Orderdate* → *Month* → *Semester* → *year*, that's why it is removed.

At the instance level, the result is shown in github. Table 1 shows the number of tuples of the original DWs (N_1 , N_2), of the merged DW (N') and the number of the common tuples (N_\cap) (tuples having

²http://tpc.org/tpc_documents_current_versions/pdf/tpc-h_v2.18.0.pdf

	Customer	Supplier	Part	Orderdate	Lineorder
N_1	11250	750	15000	1804	252689
N_2	11250	750	15000	1804	252821
N_{\cap}	8439	556	11261	1349	105345
N'	14061	944	18739	2259	400165

Table 1: Number of tuples

	Customer.Region	Supplier.Region	Orderdate.Semester
N_1	X	X	1804
N_2	X	750	X
N'	9713	846	2259
N_+	9713	96	455

Table 2: Number of attributes

the same dimension key in the original DWs). For each dimension or fact table, $N' = N_1 + N_2 - N_{\cap}$, we can thus confirm that there is no addition or loss of data. For each tuple in the original tables, we verify that the all the values are the same with the values in the merged table. We also find that there are some empty values of the attribute *Region* in the dimension *Customer* and *Supplier* and the attribute *Semester* of the dimension *Orderdate* which are completed. Table 2 shows the number of these attributes in the original DWs (N_1 , N_2) and in the merged DW (N'), we can then get the number of the completed values N_+ for these attributes. They meet the relationship $N' = N_1 + N_2 + N_+$.

5.3 Constellation schema generation

The objective of this experiment is to merge two star schemata having the same 2 dimensions (*Customer*, *Supplier*) with the same lowest level of granularity for each dimension, as well as 2 different dimensions (S_1 .*Part* and S_2 .*Date*).

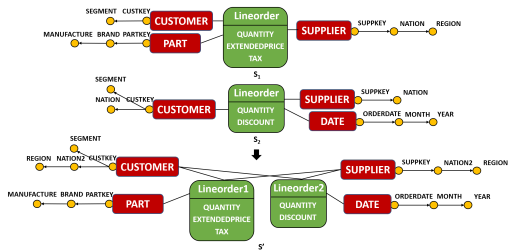


Figure 12: Constellation schema generation

At the schema level, the second test generates a constellation schema like shown in Figure 12. The run time is 32.13s. As expected, the 2 dimensions *Customer*, *Supplier* of the original DWs are merged, the other dimension and fact tables are not merged. The dimension *Customer* gains a new attribute *Region* by the merging between S_1 .*Supplier* and S_2 .*Customer*. We can see that the hierarchy *Custkey* \rightarrow *nation* of *Customer* which should be in the merged schema is deleted because there is no tuple which belongs to this hierarchy but not to *Custkey* \rightarrow *nation* \rightarrow *Region*. The hierarchy *Supkey* \rightarrow *nation* of *Supplier* is removed due to the same reason.

At the instance level, the data of experiment can be found in github. They also meet $N' = N_1 + N_2 - N_{\cap}$. There are empty values

of the attribute *Region* in the dimension *Customer* and *Supplier* which are completed which meet $N' = N_1 + N_2 + N_+$.

We got the results conforming to our expectations in the tests, we can thus conclude that our algorithms work well for the different cases discussed at both schema and instance levels.

6 CONCLUSION AND FUTURE WORK

In this paper, we define an automatic approach to merge two different star schema-modeled DWs, by merging multidimensional schema elements including hierarchies, dimensions and facts at the schema and instance levels. We define the corresponding algorithms, which consider different cases. Our algorithms are implemented and illustrated by various examples.

Since we only discuss the merging of DWs modeled as star schemata in this paper, which is only one (albeit common) possible DW design, we plan to extend our approach by adding the merging of DWs modelled as constellation schemata in the future. There may also be so-called weak attributes in DW components. Thus, we will consider them in future work. Our goal is to provide a complete approach that is integrated in our previous work concerning the automatic integration of tabular data in DWs.

ACKNOWLEDGEMENTS

The research depicted in this paper is funded by the French National Research Agency (ANR), project ANR-19-CE23-0005 BI4people (Business Intelligence for the people).

REFERENCES

- [1] Sina Ariyan and Leopoldo Bertossi. 2011. Structural Repairs of Multidimensional Databases. In *Inter. Workshop on Foundations of Data Management*, Vol. 748.
- [2] Marko Banek, Boris Vrdoljak, A. Min Tjoa, and Zoran Skočir. 2007. Automating the Schema Matching Process for Heterogeneous Data Warehouses. In *Data Warehousing and Knowledge Discovery*. 45–54.
- [3] S. Bergamaschi, M. Olaru, S. Sorrentino, and M. Vincini. 2011. Semi-automatic Discovery of Mappings Between Heterogeneous Data Warehouse Dimensions. *J. of Computing and Information Technology* (dec 2011), 38–46.
- [4] Philip A. Bernstein, Jayant Madhavan, and Erhard Rahm. 2011. Generic Schema Matching, Ten Years Later. *Proc. VLDB Endow.* 4, 11 (aug 2011), 695–701.
- [5] Elhaj Elamin, Amer Alzaidi, and Jamel Feki. 2018. A Semantic Resource Based Approach for Star Schemas Matching. *IJDMS* 10, 6 (dec 2018).
- [6] S. Anitha Elavarasi, J. Akilandeswari, and K. Menaga. 2014. A Survey on Semantic Similarity Measure. *Inter. J. of Research in Advent Technology* 2 (mar 2014).
- [7] Jamel Feki, Jihen Majdoubi, and Faiez Gargouri. 2005. A Two-Phase Approach for Multidimensional Schemas Integration. In *17th Inter. Conference on Software Engineering and Knowledge Engineering*. 498–503.
- [8] M. Kwakye, I. Kiringa, and H. L. Viktor. 2013. Merging Multidimensional Data Models: A Practical Approach for Schema and Data Instances. In *5th Inter. Conference on Advances in Databases, Data, and Knowledge Applications*.
- [9] Salvatore T. March and Alan R. Hevner. 2007. Integrated decision support systems: A data warehousing perspective. *Decis. Support Syst.* 43, 3 (apr 2007), 1031 – 1043.
- [10] Lingling Meng, Runqing Huang, and Junzhong Gu. 2013. A review of semantic similarity measures in wordnet. *IJHIT* 6 (jan 2013).
- [11] Marius-Octavian Olaru and Maurizio Vincini. 2012. A Dimension Integration Method for a Heterogeneous Data Warehouse Environment. In *Inter. Conf. on Data Communication Networking, e-Business and Optical Communication Systems*.
- [12] Christoph Quix, David Kensch, and Xiang Li. 2007. Generic Schema Merging. In *Advanced Information Systems Engineering*. 127–141.
- [13] Franck Ravat, Olivier Teste, Ronan Tournier, and Gilles Zurfluh. 2008. Algebraic and Graphic Languages for OLAP Manipulations. *Inter. J. of Data Warehousing and Mining* 4 (jan 2008), 17–46.
- [14] Oscar Romero and Alberto Abelló. 2009. A Survey of Multidimensional Modeling Methodologies. *Inter. J. of Data Warehousing and Mining* 5, 2 (apr 2009).
- [15] Riccardo Torlone. 2008. Two approaches to the integration of heterogeneous data warehouses. *Distributed and Parallel Databases* 23 (feb 2008), 69–97.

Multi-Model Data Modeling and Representation: State of the Art and Research Challenges

Pavel Čuntoš
contos@ksi.mff.cuni.cz
Charles University

Faculty of Mathematics and Physics
Prague, Czech Republic

Irena Holubová
holubova@ksi.mff.cuni.cz
Charles University

Faculty of Mathematics and Physics
Prague, Czech Republic

Martin Svoboda
svoboda@ksi.mff.cuni.cz
Charles University

Faculty of Mathematics and Physics
Prague, Czech Republic

ABSTRACT

Following the current trend, most of the well-known database systems, being relational, NoSQL, or NewSQL, denote themselves as multi-model. This industry-driven approach, however, lacks plenty of important features of the traditional DBMSs. The primary problem is a design of an optimal multi-model schema and its sufficiently general and efficient representation. In this paper, we provide an overview and discussion of the promising approaches that could potentially be capable of solving these issues, along with a summary of the remaining open problems.

CCS CONCEPTS

• **Information systems** → **Entity relationship models; Semi-structured data; Data structures; Integrity checking; Query languages for non-relational engines.**

KEYWORDS

Multi-model data, Inter-model relationships, Conceptual modeling, Logical models, Category theory

ACM Reference Format:

Pavel Čuntoš, Irena Holubová, and Martin Svoboda. 2021. Multi-Model Data Modeling and Representation: State of the Art and Research Challenges. In *25th International Database Engineering & Applications Symposium (IDEAS 2021)*, July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3472163.3472267>

1 INTRODUCTION

In recent years, the Big Data movement has broken down borders of many technologies and approaches that have so far been widely acknowledged as mature and robust. One of the most challenging issues is the *variety* of data which may be present in multiple types and formats (structured, semi-structured, and unstructured), and so conform to various models.

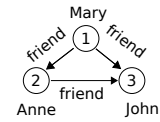
EXAMPLE 1. *Let us consider an example of a multi-model scenario in Figure 1 (partially borrowed from [17]), backing a sample enterprise management information system where individual parts of the data are intentionally stored within different logical models. In particular, the social network of customers is*

captured using a graph \mathcal{G} (blue¹). Relational table \mathcal{T} (violet) records additional information about these customers, such as their credit limit. Orders submitted by customers are stored as JSON documents in a collection \mathcal{D} (green). A wide-column table \mathcal{W} (red) then maintains the history of all orders, and, finally, a key/value mapping \mathcal{K} (brown) stores currently existing shopping carts. A cross-model query might return, e.g., friends of customers who ordered any item with a price higher than 180. □

relational table \mathcal{T}

customer	name	address	credit
1	Mary	...	3 000
2	Anne	...	2 000
3	John	...	5 000

property graph \mathcal{G}



document collection \mathcal{D}

```
{ order : 220,  
  paid : true,  
  items : [  
    { product : T1, name : toy,  
      price : 200, quantity : 2 },  
    { product : B4, name : book,  
      price : 150, quantity : 1 } ] }
```

wide-column table \mathcal{W}

customer	orders
1	[220, 230, 270, ...]
2	[10, 217]
3	[370, 214, 94, 137]

key/value pairs \mathcal{K}

customer	cart
1	product: T1, name: toy, quantity: 2 product: B4, name: book, quantity: 1
2	product: G1, name: glasses, quantity: 1 product: B2, name: book, quantity: 1
3	product: B3, name: book, quantity: 2

Figure 1: Sample multi-model scenario

While the problem of multi-model data management may seem similar to the data integration and, hence, some approaches/ideas can be re-used, the aim and motivation are different. In the multi-model world, each of the involved single-model schemas represents just a certain part of reality. The individual distinct models are chosen in order to conform to the structure of the part of the reality the best (and thus enable its optimal logical and physical representation). These parts are mutually interlinked, and together they form the whole reality. The multi-model schema can be viewed as a possible result of the data integration process, if the particular integrated sources represented distinct data models and these features needed to be preserved.

¹Individual colors are used to highlight the different models in the rest of the text.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions.acm.org.

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Association for Computing Machinery.

ACM ISBN 978-1-4503-8991-4/21/07...\$15.00

<https://doi.org/10.1145/3472163.3472267>

Depending on the number of (different) underlying database management systems (DBMSs) being involved, the so-called *multi-model systems* can be divided into single-DBMS and multi-DBMS (i.e., *polystores*). We target mainly the former group in this paper since such an approach is industry-driven rather than academic and thus more popular. Currently, there exist more than 20 representatives of single-DBMS multi-model databases, involving well-known tools from both the traditional relational and novel NoSQL or NewSQL systems, such as Oracle DB, IBM DB2, Cassandra, or MongoDB, to name just a few.

According to a recent extensive survey [16], they have distinct features and can be classified according to various criteria. The core difference is the strategy used to extend the original model to other models or to combine multiple models: such new models can be supported via the adoption of an entirely new storage strategy, extension of the original storage strategy, creation of a new interface, or even no change in the original storage strategy (which is used for trivial cases). This bottom-up approach, driven by the needs of novel applications, lacks plenty of important features of the traditional database systems [10], though.

The primary issue is designing an optimal multi-model schema and its sufficiently robust representation that can evolve efficiently and correctly with changes in user requirements. Although there exist mature and verified approaches commonly and successfully used for individual models, most of them cannot be easily extended for the multi-model data due to the contradictory features of the models (such as relational vs. hierarchical vs. graph data).

In this paper, we provide an overview and discussion of existing approaches that are close to the world of multi-model data and could be exploited for this purpose. The main contributions of this paper are:

- overview of the relevant related work for multi-model data modeling and representation (Section 2),
- introduction to category theory and its relation to multi-model data management (Section 3), and
- discussion of remaining research opportunities together with possible inspirational solutions (Section 4).

Our objective is to introduce a promising research direction for the data management community, both for researchers and practitioners. Considering the amount of available multi-model databases, the indicated target definitely has a high impact in both academia and industry, but still requires extensive research and implementation effort.

2 EXISTING APPROACHES

In multi-model scenarios, we intentionally combine several structurally different logical data models. To simplify the description and understanding, we may consider only the currently most commonly used ones: relational, aggregate-oriented (i.e., key/value, wide-column, document), and graph. Despite definitions of these data models involve at least the required data structures, other aspects can be covered as well. E.g., following Codd [7], (at least) the following three components can be assumed: (1) data structures, (2) operations on data structures, and (3) integrity constraints for operations and structures.

Pursuing the ideas of model-driven engineering [18], there are different layers of information abstraction. The *conceptual* platform-independent model (PIM) describing the problem domain is translated to one or more *logical* platform-specific models (PSMs), such as relational or semi-structured (e.g., XML), represented using a selected schema language (e.g., SQL DDL or XML Schema).

In the following sections, we will have a look at a wide range of existing approaches and demonstrate how they can be utilized when the processing of multi-model data is to be tackled. In particular, we will cover five basic aspects closely related to data modeling and representation: data design (Section 2.1), logical data representation (Section 2.2), integrity constraints (Section 2.3), knowledge reasoning (Section 2.4), and evolution management (Section 2.5).

2.1 Conceptual Layer

The purpose of the platform-independent layer is to allow us to model schemas of databases at the conceptual layer, i.e., without limiting ourselves to features of specific logical models. This means we can think of the database contents purely in terms of sets of entities, relationships, and their characteristics, and so particular data structures that will, later on, be used for the actual data representation and physical storage (as well as the implied consequences and limitations) are purposely treated as unimportant for us at this moment.

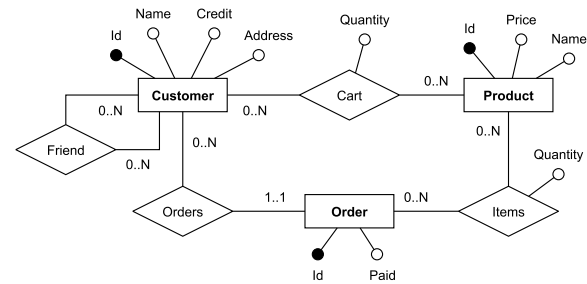


Figure 2: ER conceptual schema

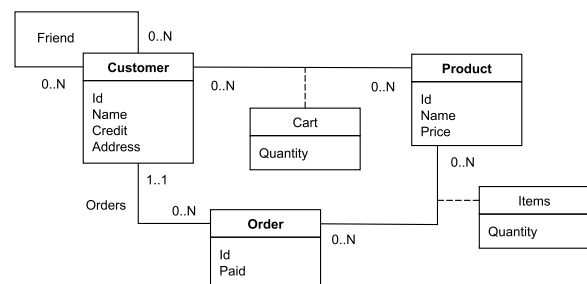


Figure 3: UML conceptual schema

There are basically two widely used traditional conceptual modeling languages: ER [5] and UML [8] (namely class diagrams). The former one exists in several notations differing not just visually, but also in particular constructs and their variations. Altogether, ER is more expressive and allows us to better grasp the complex nature

and features of the real-world entities and their relationships. On the other hand, UML is well-standardized. Unfortunately, also only too data-oriented. This means that its constructs and expressive power are not that elaborate, and so UML schemas may conceal certain details that could be significant for us (e.g., structures such as weak entity types, structured attributes, etc.).

Having a conceptual schema, it can then be transformed to a logical schema as a whole, as well as it can be transformed to multiple distinct logical schemas at a time, each covering the entire domain, or just a part of it, each mutually disjoint, connected, or more-or-less overlapping. This principle, and hence the entire idea of platform-independent modeling, apparently becomes especially important when working with data that really is, by nature, multi-model.

Our ultimate objective is the ability to work with such multi-model data in a unified way that simplifies the reality, yet still thoroughly enough captures specifics of at least the majority of widely used logical models and data formats exploited in the existing database implementations. Despite both ER and UML might seem well-suited for this purpose, there are various questions to be answered and trade-offs to be established. For example, not just the following aspects deserve attention: permitted ranges of relationship type cardinalities or attribute multiplicities (particular values or just N for the upper bounds), the semantics of these cardinalities in n -ary associations, non-trivial depths of structured attributes, the necessity of identifiers for entity types, the possibility of identifiers even for relationship types, incorporation of just selected participants in weak identifiers instead of the involved relationship types as a whole, etc.

EXAMPLE 2. Continuing with the sample multi-model data presented in Figure 1, its corresponding ER and UML schema diagrams are visualized in Figures 2 and 3. \square

2.2 Logical Layer

The logical layer of data modeling aims to provide data structures in which the actual data is represented within a particular database system. For example, we can have key/value, document, or graph models. In fact, there is a wide span of such approaches, involving academic proposals (e.g., the X-SEM model [21] for XML data), official standards, as well as models provided by the existing DBMS representatives (e.g., the formal relational model vs. its derived versions). They can be abstract and unifying or not (generic trees vs. JSON), they can cover only one underlying model or more of them at a time (property graphs vs. associative arrays), they can consider only the structure of the data or extend it with its schema or integrity constraints, or they can even be represented by the PIM directly (such as the *whiteboard-friendly* [22] Neo4j graph model).

NoSQL Abstract Model (NoAM). The first approach we start with, NoAM [1], specifies an intermediate, system-independent data representation for aggregate-oriented (i.e., key/value, wide-column, and document) NoSQL systems. The proposed methodology aims at designing a good (with regards to the performance, scalability, and consistency) representation of the data in NoSQL systems.

A NoAM database is a set of *collections* having distinct names (e.g., a wide-column table or a document collection). A collection is a set of *blocks* (i.e., aggregates), each identified by a *block key*

unique within that collection (e.g., a row in a wide-column table or a document in a collection) and being a maximal data unit for which atomic, efficient, and scalable access operations are provided. A block is a non-empty set of *entries* (e.g., table columns or document fields). An entry is a pair (e_k, e_v) , where e_k is the *entry key* (unique within its block), and e_v is (complex or scalar) *entry value*.

EXAMPLE 3. Consider the document model of our sample data in Example 1. NoAM first defines two basic representation strategies – *Entry per Aggregate Object (EAO)* and *Entry per Top-level Field (ETF)*. In EAO, document collection \mathcal{D} would be represented as depicted in Figure 4 (a). Each block corresponds to one order, block key corresponds to the order identifier (220). There is only one entry in each block, so the entry key is empty (ϵ), and the value contains the rest of the data. For ETF, the block would have two entries with keys corresponding to top-level fields, as shown in Figure 4 (b). \square

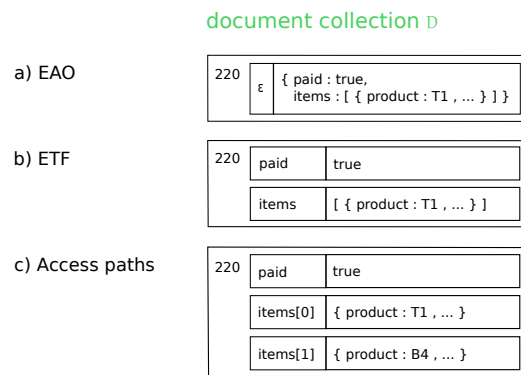


Figure 4: Strategies of NoAM

Next, data access patterns are considered using the notion of an *access path*, i.e., a location in the structure of a complex value. If v is a complex value and v' is a (possibly complex) value occurring in v , then the access path a_p for v' in v represents the sequence of steps that need to be taken to reach the component value v' in v . A complex value v can then be represented using a set of entries, whose keys are access paths in v . Each entry is expected to represent a distinct portion of the complex value v , characterized by a location in its structure.

EXAMPLE 4. Based on the access patterns, collection \mathcal{D} could be partitioned, e.g., as depicted in Figure 4 (c), where access paths to particular fields in the array are assumed. \square

The proposed good choice of aggregates and their partitioning is driven by the data access patterns of the application operations, as well as by scalability and consistency needs. In addition, to support the strong consistency of update operations, each aggregate should include all the data involved by some integrity constraints. On the other hand, aggregates should be as small as possible. Moreover, a particular set of rules for the partitioning of the data model into aggregates that reflects all these requirements is actually proposed in the paper.

Unfortunately, despite the fact that this approach considers a set of data models, though only closely related to aggregate-oriented ones, they are not considered in a combination but separately. The proposed strategies for a good aggregate design indicate how this

aim differs from the traditional SQL and NoSQL world and what should be taken into account to cover them both.

Associative Arrays. The idea of *associative arrays* [11] enables us to logically (and possibly also physically) represent all data models using a single generic data structure providing an abstraction for many classes of databases (SQL, NoSQL, NewSQL, or graph).

An associative array A (naturally corresponding to a table or matrix) is defined as a mapping from pairs of keys to values $A : K_1 \times K_2 \rightarrow \mathbb{V}$, where K_1 are row keys, K_2 column keys, both of which can be any sortable sets (integers, strings, etc.). The following rules must hold: (1) each row key and each column key is unique, and (2) there are no rows or columns that would be entirely empty. Basic operations for associative arrays are element-wise addition, element-wise multiplication, and array multiplication, which correspond, e.g., to the database operations of table union, intersection, and transformation.

relational table \mathcal{T}

	name	address	credit
1	Mary	...	3 000
2	Anne	...	2 000
3	John	...	5 000

key/value pairs \mathcal{K}

	cart
1	...
2	...
3	...

wide-column table \mathcal{W}

	01	02	03	04	...
1	220	230	270	...	
2	10	217			
3	370	214	94	137	

property graph \mathcal{G}

	1	2	3
1	0	1	1
2	0	0	1
3	0	0	0

document collection \mathcal{D}

	order	paid	items/product	items/name	...
001	220	true			
001/001			T1	toy	
001/002			B4	book	

Figure 5: Associative arrays for various models

EXAMPLE 5. As shown in Figure 5, for relational \mathcal{T} , key/value \mathcal{K} , and wide-column \mathcal{W} models, the representation in associative arrays is straightforward. For graph model \mathcal{G} , we can use any graph representation using matrices. Hierarchical data \mathcal{D} can be represented as sparse arrays by traversing the hierarchy and incrementing/appending the row counters and column keys to emit row-column-value triples. \square

Tensor Data Model (TDM). The TDM [14] permits us to view multi-model data using the notion of a *tensor*. It is often denoted as a generalized matrix: 0-order tensor is a scalar, 1-order tensor is a vector, 2-order a traditional matrix, and tensors of order 3 or higher are called *higher-order tensors*. In TDM, tensor dimensions and values are defined as a map that associates keys to values as $T : K_1 \times \dots \times K_n \rightarrow \mathbb{V}$, where K_i for $i \in \{1, \dots, n\}$ are sets of keys, $n \in \mathbb{N}$ is a dimension, and \mathbb{V} is the set of values. In addition, the tensors are named and typed. Tensor operations, analogously to operations on matrices and vectors, are multiplications, transposition, unfolding (transforming a tensor to a matrix), factorizations (decompositions), or other.

EXAMPLE 6. Relational \mathcal{T} , key/value \mathcal{K} , wide-column \mathcal{W} and graph \mathcal{G} models are mapped to tensors straightforwardly, similarly to Figure 5. Multigraphs can be modeled by a 3-order tensor where one dimension is used to specify the different types of edges. Document model \mathcal{D} is not considered in TDM. \square

2.3 Integrity Constraints

Integrity constraints representing various conditions the data must abide by form another component of the data modeling [7]. *Object Constraint Language* (OCL) [24], a part of UML, is one of the common approaches – a declarative and strongly typed language allowing to express complex integrity constraints. Such as those that would otherwise be difficult or even impossible to express using cardinalities or other constructs within the UML conceptual schemas themselves.

Thorough enumeration of all kinds of constructs provided by OCL would be beyond the scope of this paper. Therefore, let us only shortly mention some of the constructs, namely those used most frequently. Although OCL supports expressions using which, for example, one can describe pre-conditions and post-conditions for methods and operations as well as rules for initial or derived values of attributes, invariants are apparently of the highest importance. Their purpose is to define assertions that instances of the static data must satisfy all the time. Invariants are written in a form of context *type* inv *name* : *expression* (though for our purpose a bit simplified), where *name* is an optional constraint name and *type* is a name of a specific type (UML class) such that its instances should satisfy the *expression* condition. This condition can be simple or complex, involving Boolean expressions with various logical connectives, navigation operators (so that we can go through UML associations and reach their counterparties), let expressions allowing us to perform auxiliary variable assignments, calls of various functions such as *size()* on collections, as well as, in particular, simulate the existential and universal quantifiers via *exists()* and *forall()* functions.

EXAMPLE 7. The following invariant ensures that each order of any customer (in the sample data) must have at least one ordered item:

```
context Customer inv :
    self.Orders->forall( o | o.Items->size() >= 1 )    □
```

The authors of OCL pursued an objective to propose a language that would be formal enough (so that different interpretations would be avoided), yet a language with syntax still user-friendly enough (so that it could be used even without more profound mathematical skills). Although there are already approaches enabling the transformation of OCL conditions into SQL expressions for the relational model, as well as OCL itself respects the nature of multi-model data (since it resides at the platform-independent layer), its broader applicability in this context is not straightforward without appropriate support and tools. Furthermore, it is necessary to propose means how integrity constraints across models are to be represented, as well as implemented and validated, in particular.

2.4 Knowledge Reasoning

Description logics [3] form a family of formalisms for *knowledge representation*, one of the fields of artificial intelligence. It is also tightly related and widely applicable in the context of processing and modeling of RDF data, possibly enriched with RDFS schemas or OWL ontologies. The purpose of a *description languages* is to capture information about a part of the world in which we are interested. Working with the *open-world assumption*, the emphasis is put in the *reasoning* functionality via which one is able to infer

new facts that are not provided explicitly in the database, i.e., the knowledge base.

Basic building blocks are formed by *concepts* and *roles*, denoting sets of individuals and their binary relationships, respectively. Having elementary descriptions in a form of *atomic* concepts and roles, as well as the *universal* \top concept (all the individuals) and *bottom* \perp concept (none individual), more complex descriptions of concepts can be obtained inductively by using various kinds of *constructors*: $\neg A$ (atomic or general negation), $A \sqcap B$ (intersection), $A \sqcup B$ (union), $\forall R.B$ (value restriction), $\exists R.\top$ and $\exists R.B$ (limited and full existential quantifications), and, finally, $\geq nR$ and $\leq nR$ (at-least and at-most number restrictions), where A and B are concepts, R role, and $n \in \mathbb{N}$.

Depending on the supported subset of the introduced constructors, individual representatives of particular description languages are distinguished. Their expressive power varies greatly, the \mathcal{AL} (attributive language) being the minimal one of practical interest. Finding reasonable trade-offs is necessary in order to deal with tractability aspects, i.e., to ensure that reasoning and other related problems are decidable in polynomial time.

A knowledge base system consists of two components: *terminology* (*TBox*) and *assertions* (*ABox*). While the purpose of the first component is to provide a set of terminological axioms in a form of *inclusions* ($A \sqsubseteq B$) and *equalities* ($A \equiv B$, usually in a form of *definitions*, where there are only symbolic names on their left-hand sides, and each one of them is defined at most once), the latter one describes the extensional data, i.e., assertions about named individuals. Formal semantics of these assertions can be well defined using a fragment of the first-order logic, in particular by unary predicates for atomic concepts, binary predicates for roles, and more complex formulae in the case of the derived concepts. Working with the description language expressions is, however, more convenient, not just because there is no need to use variables that would otherwise be needed in the translated formulae.

More complicated expressions for derived concepts can actually be viewed as a kind of querying, yet at a layer of lower granularity focusing on individuals (entities) rather than their inner characteristics (attributes, properties, etc.), and so as if not fully exploiting the needs of multi-model data processing.

EXAMPLE 8. Assuming that *Orders*² is a role describing a mapping from customers to orders, $Q \equiv \text{Customer} \sqcap \forall \text{Friend}.(\geq 3 \text{ Orders})$ then describes customers such that all their friends have at least 3 orders, i.e., $Q^I = \text{Customer}^I \cap \{c \mid c \in \Delta^I \wedge \forall f : (c, f) \in \text{Friend}^I \Rightarrow f \in \{w \mid w \in \Delta^I \wedge |\{o \mid (w, o) \in \text{Orders}^I\}| \geq 3\}\}$, where I is the interpretation structure with the domain of individuals Δ^I . \square

Having complex concepts derived, we can perform the *instance classification* tests (whether a particular individual belongs to a given concept) or *retrievals* (acquiring a set of all individuals belonging to a given concept). Besides the granularity, another limitation is brought by the idea that only binary roles are assumed, and derivation of complex roles is usually not considered.

²We are aware of a widely adopted convention where concepts are usually named by nouns, while names of roles are derived from verbs (e.g., *makesOrder* instead of our *Orders*). However, we intentionally decided not to follow these principles and instead we use names that directly correspond to the names of individual entities as they are introduced in our multi-model scenario presented in Figure 1.

2.5 Evolution Management

No matter how well a schema of data is designed, sooner or later user requirements may change and such changes then need to be appropriately reflected in both the data structures and all related parts of the system, too.

DaemonX. In paper [27], the authors propose a *five-level evolution management framework* called *DaemonX*. For the forward-engineering design of an application, they consider a top-down approach starting from the design of a PIM and then mapping its selected parts to the respective single-model PSMs (followed by schema, operational, and extensional levels). The supported PSM models involve XML, relational, business-process, and REST. The authors focus on correct and complete propagation of changes between the models, especially when the PSM schemas overlap, and the change must be propagated correctly to all the instances. In addition, mapping to queries and respective propagation of changes to the operational level is considered, too.

EXAMPLE 9. *DaemonX* uses the classical UML class diagram (from Figure 3) for the PIM level. Parts of the PIM (possibly overlapping) are then chosen and mapped to particular PSM diagrams. In Figure 6, we depict the situation for the relational \mathcal{T} and document \mathcal{D} models. For the relational PSM (violet), the ER model (from Figure 2) is used. For the document PSM (green), the X-SEM model is used. It enables us to model the hierarchy, repetition, and other features of semi-structured data. At the schema level – not depicted in the figure – there would be the database schema of relational tables and JSON documents expressed in respective languages. \square

The idea of the framework is general and extensible, so any model which can be mapped to the common general PIM schema can be added to the framework. However, the authors do not consider inter-model links between PSM schemas, cross-model queries, nor the storage of multi-model data in a single DBMS.

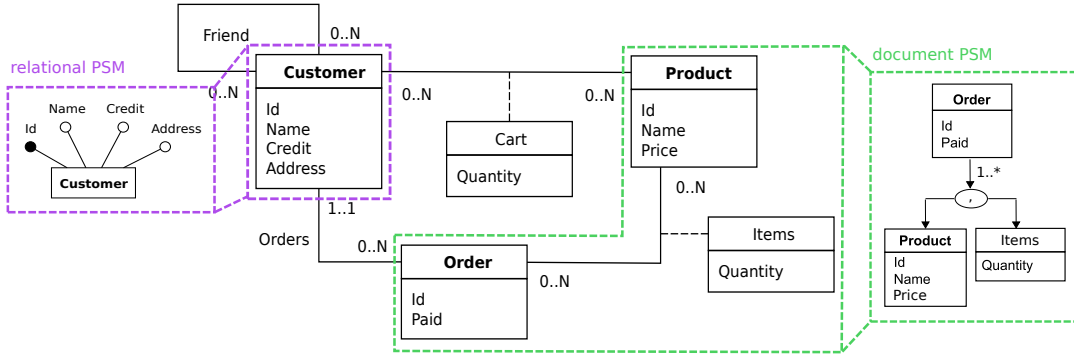
MigCast. Another approach, *MigCast* [9], also focuses on the evolution management in the multi-model world, namely aggregate-oriented NoSQL systems. It utilizes a cost model based on the characteristics of the data, expected workload, data model changes, and cloud provider pricing policies, too. While the data migration can be eager, lazy, proactive, etc., each having its (dis)advantages, *MigCast* provides an estimation of their costs and helps the users to find optimal solutions for a given application.

3 BROADER GENERALIZATION

All the approaches in the previous section can be described as promising generalizations of particular aspects of the data management towards multiple models. However, the idea of generalization can go even further and cover more data processing aspects at a time using the same formal framework.

Category theory [2] is a branch of mathematics that attempts to formalize various widely used and studied structures in terms of categories, i.e., in terms of directed labeled graphs composed from *objects* representing graph vertices, and *morphisms* (or equivalently also *arrows*), i.e., mappings between the objects, representing directed graph edges.

Formally, a category \mathbf{C} consists of a set of objects $obj(\mathbf{C})$ and a set of morphisms $mor(\mathbf{C})$, each of which is modeled and depicted as an arrow $f : A \rightarrow B$, where $A, B \in obj(\mathbf{C})$, A being treated as a

Figure 6: PIM and PSM levels in *DaemonX*

domain and B as a codomain, respectively. Whenever $f, g \in \text{mor}(\mathcal{C})$ are two morphisms $f : A \rightarrow B$ and $g : B \rightarrow C$, it must hold that $g \circ f \in \text{mor}(\mathcal{C})$, i.e., morphisms can be composed using the \circ operation and the composite $g \circ f$ must also be a morphism of the category (transitivity of morphisms is required). Moreover, \circ must satisfy the associativity, i.e., $h \circ (g \circ f) = (h \circ g) \circ f$ for any triple of morphisms $f, g, h \in \text{mor}(\mathcal{C})$, $f : A \rightarrow B$, $g : B \rightarrow C$, and $h : C \rightarrow D$. Finally, for every object A , there must exist an identity morphism 1_A such that $f \circ 1_A = f = 1_B \circ f$ for any $f : A \rightarrow B$ (obviously serving as a unit element with respect to the composition).

Although objects in real-world categories usually tend to be sets of certain items and morphisms functions between them, both objects and morphisms may actually represent abstract entities of any kind.

EXAMPLE 10. *Set* (as widely denoted) is a category where objects are arbitrary sets (not necessarily finite), and morphisms are functions between them (not necessarily injective nor surjective), together with the traditionally understood composition of functions and identities.

Having a graph $G = (V, E)$, where V is a set of vertices and $E \subseteq V \times V$ is a set of directed edges, we could derive another category where objects are the original vertices and morphisms simply the edges, composition \circ producing ordinary edges forming kind of shortcuts for the collapsed paths, and identities working as loops. Apparently, such a structure could but may not necessarily define a well-formed category, since it may happen that for any two edges (morphisms) $f = (a, b)$ and $g = (b, c) \in E$ the composite $g \circ f = (a, c) \notin E$, i.e., the composed edge may not be in the graph. \square

Category theory can also be applied to data processing (e.g., modeling, representation, transformation, querying, etc.). Actually, such proposals already exist, though not robust enough and focusing mostly just on the relational model alone (as a single-model solution only).

In the following text, we first describe an existing approach using which schemas of relational databases can be modeled (Section 3.1). Based on it, we then propose its possible extension towards the multi-model scenario to illustrate the challenges involved (Section 3.2). Returning back to the existing approaches, we then discuss how instances of relational databases (the actual data) can be modeled (Section 3.3), and, last but not least, and at a higher level of abstraction, we describe how categories of all the possible schemas or instances (Section 3.4) can be utilized for the purpose of transformations or querying (Section 3.5).

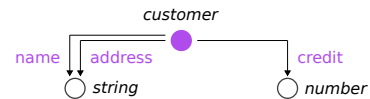
3.1 Relational Schema Category

The approach by Spivak [30] allows for a representation of schemas of relational databases in terms of categories. Suppose we have a schema of a relational database, i.e., a set of named tables, each with a compulsory single-column primary key³ and other columns (if any), possibly interlinked with foreign keys. The corresponding category \mathcal{T} (let us call it a *schema category*) will have objects of two sorts: (1) objects for tables and (2) objects for generalized data types (e.g., *number*, *string*, ...). As for the morphisms: (1) for every table t and its column c (other than the primary key and foreign key columns) of data type d , there will be a morphism $c : t \rightarrow d$, (2) for every table t , there will be an implicit identity morphism $id_t : t \rightarrow t$ allowing us to represent the primary key of t , (3) for every data type d an implicit identity morphism $id_d : d \rightarrow d$ (they will not become useful later on, but they must be there to make the category well-formed), and (4) for every foreign key from a column c of table t_1 referencing the primary key of t_2 , there will be a morphism $c : t_1 \rightarrow t_2$.

To make our description complete, let us also mention that the approach also focuses on integrity constraints [31], though we will not discuss them further.

EXAMPLE 11. The abstract schema representation of the relational table \mathcal{T} from Example 1 is depicted in Figure 7. Object *customer* represents the table itself (depicted as a full circle), objects *string* and *number* are generalized primitive types (as empty circles), and *name* : *customer* \rightarrow *string*, *address* : *customer* \rightarrow *string*, and *credit* : *customer* \rightarrow *number* are morphisms for the individual columns.

To simplify the figure, we omitted the visualization of all the involved identity morphisms: the one for the primary key of table *customer*, as well as both the identity morphisms on the involved data type objects. \square

Figure 7: Schema for table \mathcal{T}

³Unfortunately, other situations such as primary keys consisting of multiple columns or unique keys are not considered by the author. In case there is no primary key, an implicit one needs to be assumed.

3.2 Multi-Model Schema Category

Even though the presented approach considers only the relational model alone (and, unfortunately, with various limitations), perhaps, we could try to go beyond the originally proposed principles and context. In particular, and as presented in the following example, we could attempt to design a draft of an abstract schema that would attempt to cover the entire multi-model scenario.

EXAMPLE 12. Building on top of the data structures as they are defined at the logical layer in our sample multi-model scenario, the resulting schema draft could be constructed as it is depicted in Figure 8. Adopting the objects and morphisms for the relational table \mathcal{T} from the previous example, we just need to add new ones for the remaining parts of the scenario. Namely, friend morphism from graph \mathcal{G} , objects *order* and *item* and the related morphisms from documents \mathcal{D} , morphism *orders* from column family \mathcal{W} , and, finally, morphism *cart* derived from key/values in \mathcal{K} , all along with appropriate data type objects *binary* and *boolean*. \square

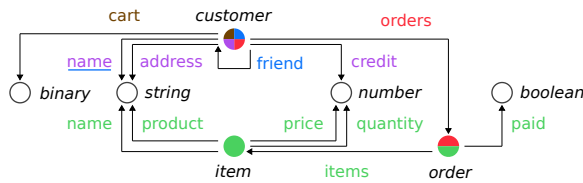


Figure 8: Schema category for the multi-model scenario

It may seem (at least from the first point of view) that categories such as these really could cover multi-model schemas. However, there are (not just) the following issues, questions, or challenges directly resulting from the draft we have just outlined. For example, how the contents of key/value pairs should be represented (whether as binary black boxes or unfolded structures), how collections and their members should be modeled (e.g., items of JSON arrays), how embedded structures should be treated (e.g., JSON subdocuments), what directions of morphisms should be selected, how duplication of morphisms and consistency of shared morphisms should be handled (e.g., names of customers in both \mathcal{T} and \mathcal{G}), how compound primary keys (or other identifiers) should be represented, how the reverse decomposition into optimal logical schemas should be performed (similarly as in NoAM), or how the knowledge available at the conceptual layer could, in general, be exploited.

3.3 Relational Instance Category

Let us now return back to the existing approaches and continue with categories using which we are able to represent particular data, i.e., instances of relational databases. In order to follow this idea, Spivak introduces a category $\text{Set}_{\mathcal{T}}$ (let us call it an *instance category*). Its purpose is to represent one particular data instance of a relational database conforming to a schema \mathcal{T} . Apparently, for every possible database content, i.e., for every possible instance, a different category $\text{Set}_{\mathcal{T}}$ will be constructed.

Similarly to a schema category \mathcal{T} itself, the newly introduced instance category $\text{Set}_{\mathcal{T}}$ will have an object for every table and every data type. The former ones are internally modeled as sets of all the actively occurring values of the corresponding primary keys, while the latter ones are internally modeled as sets of all the possible

values (domains) of the corresponding primitive data types. As for the morphisms, they are also analogous to the morphisms in the original \mathcal{T} . In particular, the purpose of the column morphisms is to allow us to assign particular column values to the individual values of the primary key, and, as a consequence, permit us to reconstruct the individual tuples (rows) of a corresponding table.

EXAMPLE 13. Having just a single relational table \mathcal{T} from Example 1 with its schema already modeled using a schema category \mathcal{T} , the instance category $\text{Set}_{\mathcal{T}}$ describing the data in our database has the following components: table object *customer* internally representing a set of values $\{1, 2, 3\}$, data type objects *string* and *number* representing the corresponding domains, column morphism *name* : *customer* \rightarrow *string* with mappings $\{(1, \text{"Mary"}), (2, \text{"Anne"}), (3, \text{"John"})\}$, morphism *address* : *customer* \rightarrow *string* defined as $\{(1, \text{"..."}), (2, \text{"..."}), (3, \text{"..."})\}$, and, finally, morphism *credit* : *customer* \rightarrow *integer* materialized as $\{(1, 3000), (2, 2000), (3, 5000)\}$. \square

3.4 Higher-Level Categories

One of the interesting features of category theory is that it easily permits us to work at different levels of abstraction. Until now, we provided several sample categories, including schema \mathcal{T} and instance $\text{Set}_{\mathcal{T}}$, the former allowing for a description of a particular database schema, the latter one description of a particular database instance. In order to be able to model more complicated database-related processes and concepts, we need to be capable of constructing categories over different categories. And for this purpose, we need to introduce the following notion of *functors*.

Assuming that \mathcal{C} and \mathcal{D} are categories, a *functor* $F : \mathcal{C} \rightarrow \mathcal{D}$ is a mapping of objects and morphisms such that the following conditions are satisfied: (1) each object $A \in \text{obj}(\mathcal{C})$ is transformed to a corresponding object $F(A) \in \text{obj}(\mathcal{D})$ (this mapping may not be bijective); (2) each morphism $f \in \text{mor}(\mathcal{C})$, $f : A \rightarrow B$ is mapped to a morphism $F(f) : F(A) \rightarrow F(B)$, specifically ensuring that $F(1_A) = 1_{F(A)}$ for any $A \in \text{obj}(\mathcal{C})$, and also guaranteeing that $F(g \circ f) = F(g) \circ F(f)$ for any suitable $f, g \in \text{mor}(\mathcal{C})$.

According to Spivak, we are now ready to introduce yet another two new categories, in particular \mathcal{T} -Schema and \mathcal{T} -Inst. While the purpose of the first one is to describe all the possible schemas of relational databases, the purpose of the second one is to describe all the possible instances of relational databases (regardless of their schemas). Let us have a look at the details at least briefly.

Category \mathcal{T} -Schema contains one object for every potentially existing relational schema, each one of them always internally modeled using a particular schema category \mathcal{T} . Morphisms between pairs of these categories, i.e., functors, allow us to model permitted changes of these schemas in terms of the introduced set of primitive schema-altering operations. Analogously, category \mathcal{T} -Inst contains one object for every potentially existing database instance, each one of them modeled as a particular instance category $\text{Set}_{\mathcal{T}}$. Morphisms (once again functors) then describe the permitted transitions between the individual instances (where not just the actual data can be changed, but schemas altered as well).

3.5 Transformations and Querying

While the approaches by Spivak presented so far only work with the relational model, an extended approach proposed by Liu et al. [15] considers the multi-model scenario. Using it, we will be able

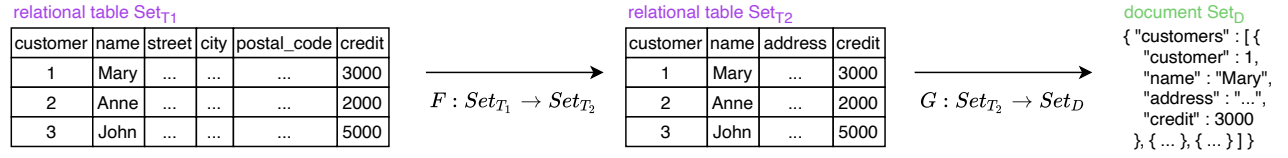


Figure 9: Sample inner and extra-model transformation

to work with multi-model data, yet only to a certain extent, because all the individual involved models are assumed to be separate (i.e., there cannot exist links or other kinds of relationships between the data in different models, which is usually not the case of real-world multi-model databases).

Supposing we have a schema category S and an instance category Set_S for every involved logical model S in our database (relational, document, etc.), we can illustrate how extra-model transformations could work, at least shortly as presented in the following example.

EXAMPLE 14. A sample transformation between tables in the relational model and an extra-model transformation between a table and a document is depicted in Figure 9. Both the tables and document contain the same data (as for their actual information content), but they differ in the structure (and, obviously, models, too).

Consider we have two instance categories for tables Set_{T_1} and Set_{T_2} and a functor $F : Set_{T_1} \rightarrow Set_{T_2}$ (describing a permitted data transformation within the relational model). We also have an instance category for document Set_D . Obviously, functor $G : Set_{T_2} \rightarrow Set_D$ can be exploited to model the intended extra-model data transformations between the relational and document model. \square

It is easy to realize that the outlined transformations can be applied at all the introduced points of view: (1) we can work with schema alterations of pairs of particular schema categories S_1 and S_2 , (2) we can study transformations of pairs of instance categories Set_{S_1} and Set_{S_2} , and, last but not least, (3) we can grasp all the possible transformations at the level of functors between pairs of schema categories as well as pairs of instance categories.

To conclude, the introduced concept of transformations can also be further used for the purpose of multi-model data querying. In particular but without details, Liu et al. [15] proposed to use (1) objects to represent individual data instances (such as tables or documents), (2) morphisms and functors to represent filtering conditions, and (3) pullbacks (as a generalization of intersection and inverse image) to represent joins of data from distinct models.

4 OPEN PROBLEMS

It may seem that there exists at least one approach for each aspect of multi-model data modeling and representation we covered in this paper. However, for each one of them, there, in fact, remain open questions that need to be solved first.

Conceptual Layer. Conceptual modeling approaches naturally support multi-model data, because they intentionally hide specifics of the individual platform-specific representations. Unfortunately, ER is not well-standardized, and UML class diagrams are concealing important details. Moreover, not just the following multi-model aspects may need to be figured out differently than the mentioned

traditional approaches assume (most likely because they are, nevertheless, closely related to the relational model at the logical layer): whether each entity type must have at least one identifier or not, whether there can be multi-valued identifiers, whether the actual values of multi-valued attributes must be mutually distinct, whether there can only be flat structured attributes or attributes with arbitrary tree structures, whether such embedded attributes may be multi-valued, whether relationship types may have their own identifiers, whether all participating entity types in n -ary relationship types really need to be involved in weak identifiers or just some of them, etc. Furthermore, a fundamental question is whether we even need to distinguish between entity types, relationship types, and attributes. Nevertheless, these aspects need to be accordingly discussed so that extended and adjusted approaches that fully respect the principles of multi-model data processing can be proposed.

Logical Layer. There exist several data structures covering the widely used data models based on the idea of mapping of sets of keys to values. However, especially the difference between the graph model and other models brings challenging questions regarding the desired natural, efficient, and unified representation. And there are other critical aspects, such as mapping of multi-model PSM(s) to PIM [34], mapping between multi-model logical and multi-model operational layers [6, 26], or logical schema inference from sample multi-model data [4, 19]. The design of the multi-model logical layer also strongly influences the other related layers as well as the mutual relations between them.

Integrity Constraints. Integrity constraints, as they are understood by OCL, represent a robust conceptual approach for describing invariants and other consistency requirements the data must conform to. Obviously, these principles may also be adopted to the multi-model scenario. However, users must be provided with practically exploitable tools. Otherwise, they will not be willing to spend time by creating such abstract descriptions. Another obstacle is that most reasoning tasks are known to be undecidable when the full expressive power of UML and OCL languages is considered. Therefore three main decidable fragments were proposed [25, 28]: (1) UML only with no OCL, (2) UML with limited OCL and no maximum cardinality constraints (OCL-Lite), and (3) UML with limited OCL with no minimum cardinality constraints (OCL_{UNIV}). The problem is that real-world UML schemas often use OCL together with min and max cardinalities, which therefore represents a limitation of the existing approaches. Fortunately, even OCL_{UNIV} can be decidable under certain assumptions [25].

Knowledge Reasoning. Because of the theoretical foundations and complexity of the knowledge representation and reasoning, the

data model must stay simple, only involving individuals, their concepts, and binary roles. Although it is possible to take multi-model data and decompose it into such units with low granularity, the idea itself breaches the very principles of multi-model data processing, where the variety, complexity, and semi-structured nature would presumably be difficult to grasp. Therefore it is perhaps questionable whether it even makes sense to deploy knowledge reasoning approaches in the context of multi-model data. Nevertheless, one can get at least inspired by the so-called OBDA (*Ontology-based Data Access*) techniques [13, 29], where the goal is to integrate heterogeneous sources of data so that they can then be accessed at a higher level of abstraction through ontologies and their mutual mappings. A different perspective is assumed by these approaches, though. While OBDA deals with issues such as data quality or process specification, treats the data from the user perspective, and focuses on the business value of the data, we are rather interested in data representation and access patterns at the logical layer in the multi-model database scenario.

Evolution Management. Evolution management is a difficult challenge even in a single-model scenario [23]. Inter-model propagation of changes and data migration bring another dimension of complexity [33]. And we also need to consider propagation of changes to operations [6, 20, 26] or storage strategies, including data migration between the models [12]. The existing (single-model or aggregate-oriented) approaches provide first steps and promising directions, but a robust, generally applicable, and extensible multi-model solution is still missing.

Broader Generalization. Category theory represents a promising framework that could bring an interesting level of further generalization. It can be used for an abstract representation of data models (both schemas as well as data instances), data and/or schema transformations, or to capture the semantics of queries and query rewriting. One of the advantages of category theory is that it easily allows us to work at different levels of abstraction. For example, we can have a category that models all the possible database instances within a particular model [32] and use it to describe schema modifications as well as data transformations like inserts, updates, deletes, etc. Nevertheless, the existing single-model approaches are not mature enough and are not in compliance with characteristics of schemas and data in real-world databases, etc.

To summarise the ideas, conceptual modeling itself, integrity constraints, as well as knowledge reasoning represent approaches that could all be placed into the conceptual layer. ER or UML modeling languages, in fact, define the meaning and the very purpose of this layer, OCL is straightforwardly built on top of it, and description logics simplify the assumed data model to just atomic individuals. On the other hand, the purpose of particular models at the logical layer is to provide data structures that are indeed used for data representation. While there are plenty of dedicated ones, both single and multi-model, such as the traditional relational and models newly revisited or introduced by the family of NoSQL database systems (key/value, wide-column, ...) or their multi-model fellows, there are also unifying approaches, such as NoAM, associative arrays, or tensors. Besides the actual data modeling and representation,

other tasks such as data transformation, querying, or evolution are tightly bound to the logical layer, too.

When pursuing the outlined vision of the unified processing of multi-model data [10], we believe that a new inter-layer between the conceptual and logical ones needs to be introduced and well established. In a top-down manner, we can start at the conceptual layer with only too abstract representations and try to modify, extend, and tailor them so that they fit the needs and specifics of the various multi-model scenarios. On the contrary, following the bottom-up direction, particular logical representations can be combined, extended, and uplifted to meet the expectations, too.

Only then the truly unified multi-model data handling will be possible. It needs to be anchored by formally solid foundations, but with particular techniques, languages and principles still user-friendly enough, i.e., not burdened with unmanageable complexities that would prevent their direct applicability and wide dissemination. Such a fully-fledged inter-layer would need to enable the unified multi-model data modeling, transformation, description of schemas, their inference, querying, evolution, or automated database tuning, to name at least a few areas.

5 CONCLUSION

The number of existing multi-model systems grows every day. However, reaching a practical and widely usable level of maturity and robustness also requires a strong formal background and generally applicable solutions. The described and justified unification is desired even because it is not likely, at least in a long-term perspective, that users will be willing to cope with a multitude of specific and often proprietary existing abstractions, languages, and techniques.

In order to accomplish the envisioned management of multi-model data, a wide range of not just the following open questions we described will, however, need to be appropriately addressed:

- *Data representation:* conceptual modeling of multi-model data, generic or unifying data structures, co-existence of multi-model and single-model scenarios, inter-model references and embedding
- *Schema design:* description of multi-model schemas, integrity constraints and their validation, data (de)normalization, schema inference from sample data, optimal schema decomposition between models
- *Unified querying:* user-friendly query language, well-defined syntax and semantics, unified processing of multi-model data, query rewriting from/to existing languages
- *Evolution management:* intra-model and inter-model schema modification, propagation of changes to data and queries, data migration between models
- *Database tuning:* autonomous model selection, integration of new models, on-the-fly data transformation, co-existence of replicas in distinct models, load balancing

ACKNOWLEDGMENTS

The work was supported by the GAČR project number 20-22276S.

REFERENCES

- [1] Paolo Atzeni, Francesca Bugiotti, Luca Cabibbo, and Riccardo Torlone. 2020. Data Modeling in the NoSQL World. *Computer Standards and Interfaces* 67 (2020), 103–149. <https://doi.org/10.1016/j.csi.2016.10.003>

- [2] Steve Awodey. 2010. *Category Theory*. Oxford university press.
- [3] Franz Baader, Diego Calvanese, Deborah McGuinness, Peter Patel-Schneider, Daniele Nardi, et al. 2003. *The description logic handbook: Theory, implementation and applications*. Cambridge university press.
- [4] Mohamed Amine Baazizi, Housseem Ben Lahmar, Dario Colazzo, Giorgio Ghelli, and Carlo Sartiani. 2017. Schema Inference for Massive JSON Datasets. In *Proc. EDBT 2017*. OpenProceedings.org, 222–233. <https://doi.org/10.5441/002/edbt.2017.21>
- [5] P.P. Chen. 1976. The Entity-Relationship Model – Toward a Unified View of Data. *ACM Transactions on Database Systems* 1, 1 (March 1976), 9–36. <https://doi.org/10.1145/320434.320440>
- [6] Martin Chytil, Marek Polák, Martin Nečaský, and Irena Holubová. 2014. Evolution of a Relational Schema and Its Impact on SQL Queries. In *IDC '14*. Vol. 511. Springer, 5–15. https://doi.org/10.1007/978-3-319-01571-2_2
- [7] E. F. Codd. 1980. Data Models in Database Management. In *1980 Workshop on Data Abstraction, Databases and Conceptual Modeling* (Pingree Park, Colorado, USA). ACM, New York, NY, USA, 112–114.
- [8] Object Management Group. 2015. *OMG Unified Modeling Language (OMG UML), version 2.5*. <http://www.omg.org/spec/UML/2.5/>
- [9] Andrea Hillenbrand, Maksym Levchenko, Uta Störl, Stefanie Scherzinger, and Meike Klettke. 2019. MigCast: Putting a Price Tag on Data Model Evolution in NoSQL Data Stores. In *SIGMOD '19* (Amsterdam, Netherlands). ACM, New York, NY, USA, 1925–1928. <https://doi.org/10.1145/3299869.3320223>
- [10] Irena Holubová, Martin Svoboda, and Jiaheng Lu. 2019. Unified Management of Multi-model Data. In *ER '19*. Springer, 439–447. https://doi.org/10.1007/978-3-030-33223-5_36
- [11] Jeremy Kepner, Julian Chaidez, Vijay Gadepally, and Hayden Jansen. 2015. Associative Arrays: Unified Mathematics for Spreadsheets, Databases, Matrices, and Graphs. *CoRR* abs/1501.05709 (2015). arXiv:1501.05709
- [12] Meike Klettke, Uta Störl, Manuel Shenavai, and Stefanie Scherzinger. 2016. NoSQL Schema Evolution and Big Data Migration at Scale. In *BigData '16*. IEEE, 2764–2774. <https://doi.org/10.1109/BigData.2016.7840924>
- [13] Roman Kontchakov, Mariano Rodríguez-Muro, and Michael Zakharyashev. 2013. *Ontology-Based Data Access with Databases: A Short Course*. Springer, Berlin, 194–229. https://doi.org/10.1007/978-3-642-39784-4_5
- [14] Eric Leclercq and Marinette Savonnet. 2019. TDM: A Tensor Data Model for Logical Data Independence in Polystore Systems. In *VLDB '18 Workshops*. Springer, 39–56. https://doi.org/10.1007/978-3-030-14177-6_4
- [15] Zhen Hua Liu, Jiaheng Lu, Dieter Gawlick, Heli Helskyaho, Gregory Pogossians, and Zhe Wu. 2019. Multi-model Database Management Systems - A Look Forward. In *VLDB '19 Workshops*. Springer, 16–29.
- [16] Jiaheng Lu and Irena Holubová. 2019. Multi-Model Databases: A New Journey to Handle the Variety of Data. *ACM Comput. Surv.* 52, 3, Article 55 (June 2019), 38 pages. <https://doi.org/10.1145/3323214>
- [17] Jiaheng Lu, Irena Holubová, and Bogdan Cautis. 2018. Multi-model Databases and Tightly Integrated Polystores: Current Practices, Comparisons, and Open Challenges. In *CIKM '18*. ACM, 2301–2302. <https://doi.org/10.1145/3269206.3274269>
- [18] J. Miller and J. Mukerji. 2003. *MDA Guide Version 1.0.1*. Object Management Group. <http://www.omg.org/docs/omg/03-06-01.pdf>.
- [19] Irena Mlynková and Martin Nečaský. 2013. Heuristic Methods for Inference of XML Schemas: Lessons Learned and Open Issues. *Informatica* 24, 4 (Oct. 2013), 577–602.
- [20] Mark Lukas Möller, Meike Klettke, Andrea Hillenbrand, and Uta Störl. 2019. Query Rewriting for Continuously Evolving NoSQL Databases. In *ER '19 (LNCS, Vol. 11788)*. Springer, 213–221. https://doi.org/10.1007/978-3-030-33223-5_18
- [21] Martin Necasky. 2007. XSEM: A Conceptual Model for XML. In *APCCM '07 - Volume 67* (Ballarat, Australia). ACS, Inc., AUS, 37–48.
- [22] Neo4j, Inc. 2020. *Graph Modeling Guidelines*. <https://neo4j.com/developer/guide-data-modeling/>
- [23] M. Nečaský, J. Klímeck, J. Malý, and I. Mlynková. 2012. Evolution and Change Management of XML-based Systems. *Elsevier* 85, 3 (2012), 683–707. <https://doi.org/10.1016/j.jss.2011.09.038>
- [24] Object Management Group. 2014. Object Constraint Language (OCL), version 2.4. <https://www.omg.org/spec/OCL/2.4/PDF>
- [25] Xavier Oriol and Ernest Teniente. 2017. OCL UNIV: Expressive UML/OCL Conceptual Schemas for Finite Reasoning. In *Conceptual Modeling*. Springer, 354–369. https://doi.org/10.1007/978-3-319-69904-2_28
- [26] M. Polák, I. Mlynková, and E. Pardede. 2013. XML Query Adaptation as Schema Evolves. In *ISD '12*. Springer Science+Business Media, Prato, Italy, 401–416. https://doi.org/10.1007/978-1-4614-7540-8_31
- [27] Marek Polák, Martin Nečaský, and Irena Holubová. 2013. DaemonX: Design, Adaptation, Evolution, and Management of Native XML (and More Other) Formats. In *IITWAS '13* (Vienna, Austria). ACM, New York, NY, USA, 484–493. <https://doi.org/10.1145/2539150.2539159>
- [28] Anna Queralt, Alessandro Artale, Diego Calvanese, and Ernest Teniente. 2012. OCL-Lite: A Decidable (Yet Expressive) Fragment of OCL. In *DL '12*. CEUR-WS.org.
- [29] Mariano Rodríguez-Muro, Roman Kontchakov, and Michael Zakharyashev. 2013. Ontology-Based Data Access: Ontop of Databases. In *The Semantic Web - ISWC 2013*. Springer, Berlin, 558–573. https://doi.org/10.1007/978-3-642-41335-3_35
- [30] David I. Spivak. 2009. Simplicial Databases. *CoRR* abs/0904.2012 (2009). arXiv:0904.2012
- [31] David I Spivak. 2012. Functorial Data Migration. *Information and Computation* 217 (2012), 31–51.
- [32] David I Spivak and Ryan Wisnesky. 2015. Relational Foundations for Functorial Data Migration. In *DBPL '15*. ACM, 21–28.
- [33] Michal Vavrek, Irena Holubová, and Stefanie Scherzinger. 2019. MM-evolver: A Multi-model Evolution Management Tool. In *EDBT 2019*. OpenProceedings.org, 586–589. <https://doi.org/10.5441/002/edbt.2019.62>
- [34] Ales Wojnar, Irena Mlynkova, and Jiri Dokulil. 2010. Structural and Semantic Aspects of Similarity of Document Type Definitions and XML Schemas. *Information Sciences* 180, 10 (2010), 1817 – 1836. <https://doi.org/10.1016/j.ins.2009.12.024>

ArchaeoDAL: A Data Lake for Archaeological Data Management and Analytics

Pengfei Liu
Sabine Loudcher
Jérôme Darmont

pengfei.liu@eric.univ-lyon2.fr
sabine.loudcher@univ-lyon2.fr
jerome.darmont@univ-lyon2.fr
Université de Lyon, Lyon 2, UR ERIC
Lyon, France

Camille Nous
camille.nous@cogitamus.fr
Laboratoire Cogitamus
France

ABSTRACT

With new emerging technologies, such as satellites and drones, archaeologists collect data over large areas. However, it becomes difficult to process such data in time. Archaeological data also have many different formats (images, texts, sensor data) and can be structured, semi-structured and unstructured. Such variety makes data difficult to collect, store, manage, search and analyze effectively. A few approaches have been proposed, but none of them covers the full data lifecycle nor provides an efficient data management system. Hence, we propose the use of a data lake to provide centralized data stores to host heterogeneous data, as well as tools for data quality checking, cleaning, transformation and analysis. In this paper, we propose a generic, flexible and complete data lake architecture. Our metadata management system exploits goldMEDAL, which is the most generic metadata model currently available. Finally, we detail the concrete implementation of this architecture dedicated to an archaeological project.

CCS CONCEPTS

• **Computer systems organization** → *Data flow architectures*; • **Information systems** → *Data warehouses*.

KEYWORDS

Data lake architecture, Data lake implementation, Metadata management, Archaeological data, Thesaurus

ACM Reference Format:

Pengfei Liu, Sabine Loudcher, Jérôme Darmont, and Camille Nous. 2021. ArchaeoDAL: A Data Lake for Archaeological Data Management and Analytics. In *25th International Database Engineering & Applications Symposium (IDEAS 2021)*, July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 11 pages. <https://doi.org/10.1145/3472163.3472266>

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-8991-4/21/07...\$15.00
<https://doi.org/10.1145/3472163.3472266>

1 INTRODUCTION

Over the past decade, new forms of data such as geospatial data and aerial photography have been included in archaeology research [8], leading to new challenges such as storing massive, heterogeneous data, high-performance data processing and data governance [4]. As a result, archaeologists need a platform that can host, process, analyze and share such data.

In this context, a multidisciplinary consortium of archaeologists and computer scientists proposed the HyperThesau project¹, which aims at designing a data management and analysis platform. HyperThesau has two main objectives: 1) the design and implementation of an integrated platform to host, search, analyze and share archaeological data; 2) the design of an archaeological thesaurus taking the whole data lifecycle into account, from data creation to publication.

Classical data management solutions, i.e., databases or data warehouses, only manage previously modeled structured data (schema-on-write approach). However, archaeologists need to store data of all formats and they may discover the use of data over time. Hence, we propose the use of a data lake [2], i.e., a scalable, fully integrated platform that can collect, store, clean, transform and analyze data of all types, while retaining their original formats, with no predefined structure (schema-on-read approach). Our data lake, named ArchaeoDAL, provides centralized storage for heterogeneous data and data quality checking, cleaning, transformation and analysis tools. Moreover, by including machine learning frameworks into ArchaeoDAL, we can achieve descriptive and predictive analyses.

Many existing data lake solutions provide architecture and/or implementation, but few include a metadata management system, which is nevertheless essential to avoid building a so-called data swamp, i.e., an unexploitable data lake [6, 12]. Moreover, none of the existing metadata management systems can provide all the needed metadata features we need. For example, in archaeology, thesauri are often used for organizing and searching data. Therefore, the metadata system must allow users to define one or more thesauri, associate data with specific terms and create relations between terms, e.g., synonyms and antonyms. Thus, we conclude that existing data lake architectures, including metadata systems, are not generic, flexible and complete enough for our purpose.

To address these problems, we propose in this paper a *generic, flexible and complete* data lake architecture. Moreover, our metadata

¹<https://imu.universite-lyon.fr/projet/hyperthesau-hyper-thesaurus-et-lacs-de-donnees-fouiller-la-ville-et-ses-archives-archeologiques-2018/>

model exploits and enriches goldMEDAL, which is the most generic metadata model currently available [13]. To illustrate the flexibility and completeness of ArchaeoDAL’s architecture, we provide a concrete implementation dedicated to the HyperThesau project. With a fully integrated metadata management and security system, we can not only ensure data security, but also track all data transformations.

The remainder of this paper is organized as follows. In Section 2, we review and discuss existing data lake architectures. In Section 3, we present ArchaeoDAL’s abstract architecture, implementation and deployment. In Section 4, we present two archaeological application examples. In Section 5, we finally conclude this paper and present future works.

2 DATA LAKE ARCHITECTURES

The concept of data lake was first introduced by Dixon [2] in association with the Hadoop file system, which can host large heterogeneous data sets without any predefined schema. Soon after, the data lake concept was quickly adopted [3, 10]. With the growing popularity of data lakes, many solutions were proposed. After studying them, we divide data lake architectures into two categories: 1) data storage-centric architecture; 2) data storage and processing-centric architecture.

2.1 Data Storage-Centric Architectures

In the early days, a data lake was viewed as a central, physical storage repository for any type of raw data, aiming for future insight. In this line, Inmon proposes an architecture that organizes data by formats, in so-called data ponds [6]. The raw data pond is the place where data first enters the lake. The analog data pond stores data generated by sensors or machines. The application data pond stores data generated by applications. Finally, the textual data pond stores unstructured, textual data.

Based on such zone solutions, Gorelik proposes that a common data lake architecture includes four zones [5]: a landing zone that hosts raw ingested data; a gold zone that hosts cleansed and enriched data; a work zone that hosts transformed, structured data for analysis; and a sensitive zone that hosts confidential data. Bird also proposes a similar architecture [1]. Such architectures organize data with respect to how deeply data are processed and security levels.

The advantage of storage-centric architectures is that they provide a way to organize data inside a data lake by default. However, the predefined data organization may not satisfy the requirements of all projects. For example, HyperThesau needs to store data from different research entities. Thus, one requirement is to organize data by research entities first. Yet, the bigger problem of storage-centric architectures is that they omit important parts of a data lake, e.g., data processing, metadata management, etc.

2.2 Data Storage and Processing-Centric Architectures

With the evolution of data lakes, they have been viewed as platforms, resulting in more complete architectures. Alrehamy and Walker propose a “Personal Data Lake” architecture that consists of five components [15], which addresses data ingestion and metadata

management issues. However, it transforms data into a special JSON object that stores both data and metadata. By changing the original data format, this solution contradicts the data lake philosophy of conserving original data formats.

Pankaj and Tomcy propose a data lake architecture based on the Lambda architecture (Figure 1) that covers almost all key stages of the data life cycle [14]. However, it omits metadata management and security issues. Moreover, not all data lakes need near real-time data processing capacities.

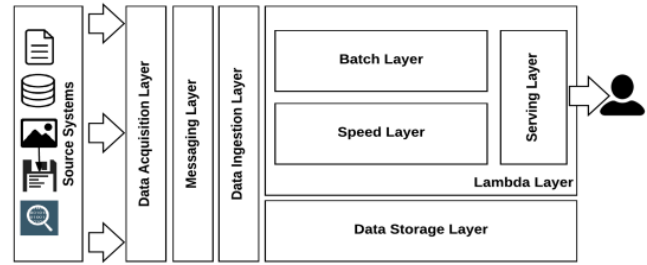


Figure 1: Lambda data lake architecture [14]

Mehmood et al. propose an interesting architecture consisting of four layers: a data ingestion layer that acquires data for storage; a data storage layer; a data exploration and analysis layer; a data visualization layer [9]. This architecture is close to our definition of a data lake, i.e., a fully integrated platform to collect, store, transform and analyze data for knowledge extraction. Moreover, Mehmood et al. provide an implementation of their architecture. However, although they mention the importance of metadata management, they do not include a metadata system in their architecture. Eventually, data security is not addressed and data visualization is the only proposed analysis method. Raju et al. propose an architecture that is similar to Mehmood et al.’s [11]. They essentially use a different tool-set to implement their approach and also omit to take metadata management and data security into account.

2.3 Discussion

In our opinion, a data lake architecture must be *generic*, *flexible* and *complete*. Genericity implies that the architecture must not rely on any specific tools nor frameworks. Flexibility means that users must be able to define their own ways of organizing data. Completeness means that not only functional features (e.g., ingestion, storage, analysis, etc.) must be handled, but also non-functional features (e.g., data governance and data security). Table 1 provides an evaluation of seven data lake architectures with respect to these three properties.

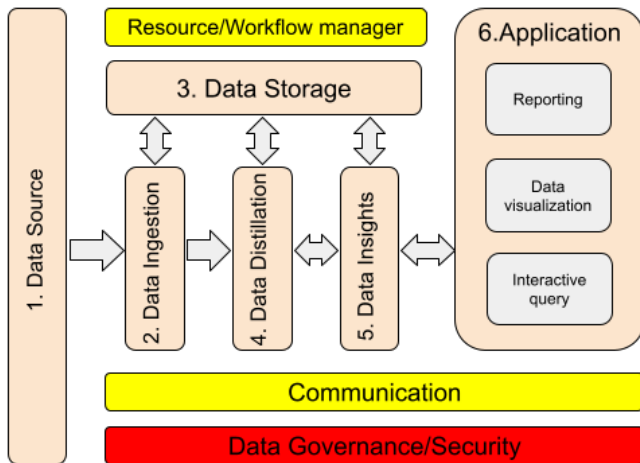
The solutions by Mehmood et al. and Raju et al. are not generic because their architecture heavily relies on certain tools. The zone architectures by Inmon, Gorelik and Bird are not flexible, because they force a specific data organization. Finally, Alrehamy and Walker’s platform is the only complete architecture that addresses data governance and security, but is not a canonical data lake.

Table 1: Comparison of data lake architectures

Architecture	Generic	Flexible	Complete
Alrehamy and Walker (2015)	✓		✓
Inmon (2016)	✓		
Pankaj and Tomcy (2017)	✓	✓	
Raju et al. (2018)		✓	
Bird (2019)	✓		
Gorelik (2019)	✓		
Mehmood et al. (2019)		✓	

3 ARCHAEODAL’S ARCHITECTURE, IMPLEMENTATION AND DEPLOYMENT

In this section, we propose a generic, flexible abstract data lake architecture that covers the full data lifecycle (Figure 2) and contains eleven layers. The orange layers (from layer 1 to layer 6) cover the full data lifecycle. After data processing in these layers, data become clearer and easier to use for end-users. The yellow layers cover non-functional requirements. After the definition of each layer, we present how each layer is implemented in our current ArchaeoDAL instance. As this instance is dedicated to the project HyperThesau, it does not cover all the features of the abstract architecture. For example, real-time data ingestion and processing are not implemented. However, real-time or near real-time data ingestion and processing feature can be achieved by adding tools such as Apache Storm² or Apache Kafka³ in the data ingestion and data insights layers.

**Figure 2: ArchaeoDAL’s architecture**

3.1 Data Source Layer

In the data source layer, we gather the basic properties of data sources, e.g., volume, format, velocity, connectivity, etc. Based on these properties, data engineers can determine how to import data

²<https://storm.apache.org/>

³<https://kafka.apache.org/>

into the lake. If metadata are required, data engineers must also find the best fitting metadata model to govern input data.

In our instance of ArchaeoDAL, we have data sources such as relational databases and various files stored in the archaeologists’ personal computers.

3.2 Data Ingestion Layer

The data ingestion layer provides a set of tools that allow users to perform batch or real-time data ingestion. Based on the data source properties that are gathered in the data source layer, data engineers can choose the right tools and plans to ingest data into the lake. They must also consider the capacity of the data lake to avoid data loss, especially for real-time data ingestion. During ingestion, metadata provided by the data sources, e.g., the name of excavation sites or instruments, must be gathered as much as possible. After data are loaded into the lake, we may lose track of data sources. It is indeed more difficult to gather this kind of metadata without knowledge about data sources.

ArchaeoDAL’s implementation exploits Apache Sqoop⁴ to ingest structured data and Apache Flume⁵ to ingest semi-structured and unstructured data. Apache Sqoop can efficiently transfer bulk data from structured data stores such as relational databases. Apache Flume is a distributed service for efficiently collecting, aggregating and moving large amounts of data. For one-time data ingestion, our instance provides Sqoop scripts and a web interface to ingest bulk data. For repeated data loading, we developed Flume agents to achieve automated data ingestion.

3.3 Data Storage Layer

The data storage layer is the core layer of a data lake. It must have the capacity to store all data, e.g., structured, semi-structured and unstructured data, in any format.

ArchaeoDAL’s implementation uses the Hadoop Distributed File System⁶ (HDFS) to store ingested data, because HDFS stores data on commodity machines and provides horizontal scalability and fault tolerance. As a result, we do not need to build large clusters. We just add nodes when data volume grows. To better support the storage of structured and semi-structured data, we add two tools: Apache Hive⁷ to store data with explicit data structures and Apache HBase⁸ that is a distributed, versioned, column-oriented database that provides better semi-structured data retrieval speed.

3.4 Data Distillation Layer

The data distillation layer provides a set of tools for data cleaning and encoding formalization. Data cleaning refers to eliminating errors such as duplicates and type violations, e.g., a numeric column contains non-numeric values. Data encoding formalization refers to converting various data and character encoding, e.g., ASCII, ISO-8859-15, or Latin-1, into a unified encoding, e.g., UTF-8, which covers all language symbols and graphic characters.

⁴<https://sqoop.apache.org/>

⁵<https://flume.apache.org/>

⁶<https://hadoop.apache.org/>

⁷<https://hive.apache.org/>

⁸<https://hbase.apache.org/>

ArchaeoDAL’s implementation uses Apache Spark⁹ to clean and transform data. We developed a set of Spark programs that can detect duplicates, NULL values and type violations. Based on the percentage of detected errors, we can hint at data quality.

3.5 Data Insights Layer

The data insights layer provides a set of tools for data transformation and exploratory analysis. Data transformation refers to transforming data from one or diverse sources into specific models that are ready to use for data application, e.g., reporting and visualization. Exploratory data analysis refers to the process of performing initial investigations on data to discover patterns, test hypotheses, eliminate meaningless columns, etc. Transformed data may also be persisted in the data storage layer for later reuse.

ArchaeoDAL’s implementation resorts to Apache Spark to perform data transformation and exploratory data analysis. Spark also provides machine learning libraries that allow developers to perform more sophisticated exploratory data analyses.

3.6 Data Application Layer

The data application layer provides applications that allow users to extract value from data. For example, a data lake may provide an interactive query system to do descriptive and predictive analytics. It may also provide tools to produce reports and visualize data.

In ArchaeoDAL’s implementation, we use a Web-based notebook, Apache Zeppelin¹⁰, as the front end. Zeppelin connects to the data analytics engine Apache Spark that can run a complex directed acyclic graph of tasks for processing data. Our notebook interface supports various languages and their associated Application Programming Interfaces (APIs), e.g., R, Python, Java, Scala and SQL. It provides a default data visualization system that can be enriched by Python or R libraries.

ArchaeoDAL also provides a web interface that helps users download, upload or delete data.

3.7 Data Governance Layer

The data governance layer provides a set of tools to establish and execute plans and programs for data quality control [7]. This layer is closely linked to the data storage, ingestion, distillation, insights and application layers to capture all relevant metadata. A key component of the data governance layer is a metadata model [12].

3.7.1 Metadata Model. In ArchaeoDAL, we adopt goldMEDAL [13], which is modeled at the conceptual (formal description), logical (graph model) and physical (various implementations) levels. goldMEDAL features four main metadata concepts (Figure 3): 1) *data entities*, i.e., basic data units such as spreadsheet tables or textual documents; 2) *groupings* that bring together data entities w.r.t. common properties in *groups*; 3) *links* that associate either data entities or groups with each other; and 4) *processes*, i.e., transformations applied to data entities that produce new data entities. All concepts bear metadata, which make goldMEDAL the most generic metadata model in the literature, to the best of our knowledge.

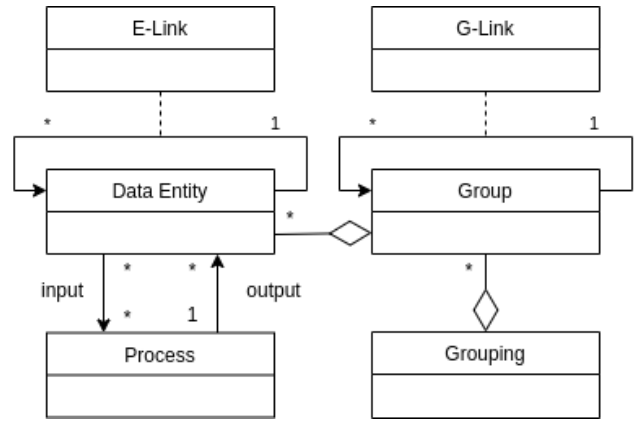


Figure 3: goldMEDAL’s concepts [13]

However, we encountered a problem of terminology variation when creating metadata. goldMEDAL does indeed not provide explicit guidance for metadata creation. This can lead to consistency and efficiency problems. For example, when users create data entities, they have their own way of defining attributes, i.e., key/value pairs that describe the basic properties of data. Without a universal guideline or template, every user can invent his own way. The number, name and type of attributes can be different. As a result, without explicit guidelines or templates, it may become quite difficult to retrieve or search metadata.

Thus, we enrich goldMEDAL with a new concept, *data entity type*, which explicitly defines the number, names and types of the attributes in a data entity.

All data entity types form a type system that specifies how metadata describe data inside the lake. Since each and every data lake has specific requirements on how to represent data to fulfill domain-specific requirements, metadata must contain adequate attributes. Thus, we need to design a domain-specific type system for each domain. For example, in the HyperThesau project, users need not only semantic metadata to understand the content of data, but also geographical metadata to know where archaeological objects are discovered. As a result, the type system is quite different from other domains.

In sum, the benefits of having a data entity type system include: 1) *consistency*, a universal definition of metadata can avoid terminology variations that may cause data retrieval problems; 2) *flexibility*, a domain-specific type system helps define specific metadata for requirements in each use case; 3) *efficiency*, with a given metadata type system, it is easy to write and implement search queries. Because we know in advance the names and types of all metadata attributes, we can filter data with metadata predicates such as `upload_date > 10/02/2016`.

3.7.2 Thesaurus Modeling. Although thesauri, ontologies and taxonomies can definitely be modeled with goldMEDAL, its specifications do not provide details on how to conceptually and logically model such semantic resources, while we especially need to manage thesauri in the HyperThesau project.

⁹<https://spark.apache.org/>

¹⁰<https://zeppelin.apache.org/>

A thesaurus consists of a set of *categories* and *terms* that help regroup data. A category may have one and only one parent. A category without a parent is called the root category. A category may have one or more children. The child of a category is called *subcategory* or *term*. A term is a special type of category that has no child but must have a parent category. A term may have relationships with other terms (related words, synonyms, antonyms, etc.).

Fortunately, categories and terms can easily be modeled as data entities and structured with labeled links, with labels defined as link metadata. It is also easy to extend this thesaurus model to represent ontologies or taxonomies.

3.7.3 Data Governance Implementation. The HyperThesau project does not require sophisticated data governance tools to fix decision domains and decide who takes decisions to ensure effective data management [7]. Thus, ArchaeoDAL's data governance layer implementation only focuses on how to use metadata to govern data inside the lake. We use Apache Atlas¹¹, which is a data governance and metadata management framework, to implement our extended version of goldMEDAL (Section 3.7.1). With these metadata, we build a data catalog that allows searching and filtering data through different metadata attributes, organize data with user-defined classifications and the thesaurus and trace data lineage.

3.8 Data Security Layer

The data security layer provides tools to ensure data security. It should ensure the user's authenticity, data confidentiality and integrity.

ArchaeoDAL's implementation orchestrates more than twenty tools and frameworks, most of which have their own authentication system. If we used their default authentication systems, a given user could have twenty login and password pairs. But even if a user decided to use the same login and password for all services, s/he would have to change it twenty times in case of need. To avoid this, we have deployed an OpenLDAP server as a centralized authentication server. All services connect to the centralized authentication server to check user login and password. The access control system consists of two parts. First, we need to control the access to each service. For example, when a user wants to create a new thesaurus, Atlas needs to check whether this user has the right to. Second, we need to control the access to data in the lake. A user may access data via different tools and the authorization answer of these tools must be uniform. We use Apache Ranger¹² to set security policy rules for each service. For data access control, we implement a role-based access control (RBAC) system by using the Hadoop group mapping service.

Data security is enforced by combining the three systems. For example, a user wants to view data from a table stored in Hive. S/he uses his/her login and password to connect to the Zeppelin notebook. This login and password are checked by the OpenLDAP server. After login, s/he runs a SQL query that is then submitted to Hive. Before query execution, Hive sends an authorization request to Ranger. If Ranger allows the user to run the query, it starts and retrieves data with the associated user credentials. Then, HDFS

checks whether the associated user credentials have the right to access data. If a user does not have the right to read data, an access denied exception is produced.

3.9 Workflow Manager Layer

The workflow manager layer provides tools to automate the flow of data processes. This layer is optional.

For now, we have not identified any repeated data processing tasks in the HyperThesau project. As a result, we have not implemented this layer. However, it can easily be implemented by integrating tools such as Apache Airflow¹³ or Apache NiFi¹⁴.

3.10 Resource Manager Layer

As data lakes may involve many servers working together, the resource manager layer is responsible for negotiating resources and scheduling tasks on each server. This layer is optional, because a reasonably small data lake may be implemented on one server only.

As ArchaeoDAL rests on a distributed storage and computation framework, a resource manager layer is mandatory. We use YARN¹⁵, since this resource manager can not only manage the resources in a cluster, but also schedule jobs.

3.11 Communication Layer

The communication layer provides tools that allow other layers, e.g., data application, data security and data governance, to communicate with each other. It must provide both synchronous and asynchronous communication capability. For example, a data transformation generates new metadata that are registered by the data governance layer. Metadata registration should not block the data transformation process. Therefore, the data insights and governance layers require asynchronous communication. However, when a user wants to visualize data, the data application and security layers require synchronous communication, since it is not desirable to have a user read data before the security layer authorizes the access.

ArchaeoDAL's implementation uses Apache Kafka, which provides a unified, high-throughput, low-latency platform for handling real-time data feeds. Kafka can connect by default many frameworks and tools, e.g., Sqoop, Flume, Hive and Spark. It also provides both synchronous and asynchronous communication.

3.12 ArchaeoDAL's Deployment

We have deployed ArchaeoDAL's implementation in a self-hosted cloud. The current platform is a cluster containing 6 virtual machines, each having 4 virtual cores, 16 GB of RAM and 1 TB of disk space. We use Ambari¹⁶ to manage and monitor the virtual machines and installed tools. The current platform allows users to ingest, store, clean, transform, analyze, visualize and share data by using different tools and frameworks.

ArchaeoDAL already hosts the data of two archaeological research facilities, i.e., Bibracte¹⁷ and Artefacts¹⁸. Artefacts currently amounts to 20,475 records and 180,478 inventoried archaeological

¹¹<https://atlas.apache.org/>

¹²<https://ranger.apache.org/>

¹³<https://airflow.apache.org/>

¹⁴<https://nifi.apache.org/>

¹⁵<https://yarnpkg.com/>

¹⁶<https://ambari.apache.org/>

¹⁷<http://www.bibracte.fr/>

¹⁸<https://artefacts.mom.fr/>

objects. Bibracte currently contains 114 excavation site reports that contain 30,106 excavation units and 83,328 inventoried objects. We have imported a thesaurus developed by our partner researchers from the *Maison de l'Orient et de la Méditerranée*¹⁹, who study ancient societies in all their aspects, from prehistory to the medieval world, in the Mediterranean countries, the Near and Middle-East territories. This thesaurus implements the ISO-25964 norm. A dedicated thesaurus by the linguistic expert of the HyperThesau project is also in the pipe. With the help of our metadata management system, users can associate data with the imported thesaurus, and our search engine allows users to search and filter data based on the thesaurus.

4 APPLICATION EXAMPLES

In this section, we illustrate how ArchaeoDAL supports users throughout the archaeological data lifecycle, via metadata. We also demonstrate the flexibility and completeness of ArchaeoDAL (Section 2.3).

4.1 Heterogeneous Archaeological Data Analysis

In this first example, we show how to analyze heterogeneous archaeological data, how to generate relevant metadata during data ingestion and transformation, and how to organize data flexibly via the metadata management system.

The Artefacts dataset consists of a SQL database of 32 tables and a set of files that stores detailed object descriptions as semi-structured data. This dataset inventories 180,478 objects.

The data management system is implemented with Apache Atlas (Section 3.7.3) and provides three ways to ingest metadata: 1) pre-coded atlas hook (script), 2) self-developed atlas hook and 3) REpresentational State Transfer (REST) API.

4.1.1 Structured Data Ingestion and Metadata Generation. To import data from a SQL database, we use a hook dedicated to Sqoop that can generate the metadata of the imported data and ingest them automatically. A new database is created in ArchaeoDAL and its metadata is generated and inserted into the Atlas instance (Figure 4). Figure 5 shows an example of table metadata inside Artefacts.

4.1.2 Semi-structured Data Ingestion and Metadata Generation. We provide three ways to ingest semi-structured data. The simplest way is to use Ambari's Web interface (Figure 6). The second way is to use the HDFS command-line client.

The third way is to use a data ingestion tool. ArchaeoDAL provides a tool called Flume. The Flume agent can monitor any file system of any computer. When a file is created on the monitored file system, the Flume agent will upload it to ArchaeoDAL automatically.

Now that we have uploaded the required files into ArchaeoDAL, we need to generate and insert the metadata into Atlas. Since Atlas does not provide a hook for HDFS, we develop our own²⁰. This hook is triggered by the HDFS create, update and delete events. For example, the upload action generates a file creation event in HDFS, which in turn triggers our metadata generation hook (Listing 1).

¹⁹<https://www.mom.fr/>

²⁰<https://github.com/pengfei99/AtlasHDFSHook>

After the hook has uploaded metadata into Atlas, it can be visualized (Figure 7).

```

1 { "entities": [ {
2     "typeName": "hdfs_path",
3     "createdBy": "pliu",
4     "attributes": {
5         "qualifiedName": "hdfs://lin02.udl.org:9000/
6         0/HyperThesau/Artefacts/object-168.txt",
7         "name": "object-168.txt",
8         "path": "hdfs://lin02.udl.org:9000/
9         HyperThesau/Artefacts",
10        "user": "pliu",
11        "group": "artefacts",
12        "creation_time": "2020-12-29",
13        "owner": "pliu",
14        "numberOfReplicas": 0,
15        "fileSize": 36763,
16        "isFile": true
17    } } ] }
```

Listing 1: Sample generated metadata from a HDFS file

We also developed an Atlas API in Python²¹ to allow data engineers to generate and insert metadata into Atlas more easily. As Amazon S3 is the most popular cloud storage, we also developed an Atlas S3 hook²².

4.1.3 Data Transformation and Metadata Generation for Data Lineage Tracing. Once Artefacts data are ingested into ArchaeoDAL, let us imagine that an archaeologist wants to link the detailed description of objects (stored in table *objects*) with their discovery and storage locations. The first step is to convert the semi-structured object descriptions into structured data. We developed a simple Spark Extract, Transform and Load (ETL) script for this sake. Then, we save the output structured data into Hive tables *location* (discovery location) and *musee* (the museum where objects are stored).

Eventually, we join the three tables into a new table called *objects_origin* that contains the objects' descriptions and their discovery and storage locations.

Thereafter, *objects_origin*'s metadata can be gathered into Atlas with the help of the default Hive hook²³ and a Spark hook developed by Hortonworks²⁴. All Hive and Spark data transformations are tracked and all relevant metadata are pushed automatically into Atlas. Figures 8 and 9 show table *objects_origin*'s metadata and lineage, respectively.

4.1.4 Flexible Data Organization. As we mentioned in Section 2.3, existing data lake solutions do not allow users to define their own ways of organizing data, while ArchaeoDAL users should. Moreover, ArchaeoDAL must allow multiple data organizations to coexist. For example, let us define four different ways to organize data: 1) by maturity, e.g., raw data vs. enriched data; 2) by provenance, e.g., Artefacts and Bibracte; 3) by confidentiality level, e.g., strictly confidential, restricted or public; and 4) by year of creation.

²¹<https://pypi.org/project/atlaspyapi/>

²²<https://pypi.org/project/atlass3hook/>

²³<https://atlas.apache.org/1.2.0/Hook-Hive.html>

²⁴<https://github.com/hortonworks-spark/spark-atlas-connector>

The screenshot shows the Apache Atlas web interface. On the left is a sidebar with navigation tabs: SEARCH, CLASSIFICATION, and GLOSSARY. The SEARCH tab is active, showing search filters for Type, Classification, Term, and Text. The main panel displays the details for the entity 'artefacts (hive_db)'. It includes a 'Classifications' section with 'Artefacts' selected, a 'Term' field, and a 'Properties' tab. The 'Properties' tab shows a table of key-value pairs for the database metadata.

Key	Value
clusterName	HyperThesau
location	hdfs://lin02.udl.org:8020/warehouse/tablespace/managed/hive/artefacts.db
name	artefacts
owner	hive
ownerType	USER
parameters	{}
qualifiedName	artefacts@HyperThesau

Figure 4: Database metadata

The screenshot shows the Apache Atlas web interface for the entity 'location (hive_table)'. It includes a 'Classifications' section with 'Artefacts' selected, a 'Term' field, and tabs for Properties, Lineage, Relationships, Classifications, Audits, and Schema. The 'Properties' tab is active, showing a table of key-value pairs for the table metadata. A 'Show Empty Values' toggle is visible in the top right of the table.

Key	Value
columns (6)	<pre>id id_pays id_departement id_commune id_lieu_dits</pre>
comment	Imported by sqoop on 2019/11/20 18:10:10
createTime	Wed Nov 20 2019 18:10:14 GMT+0100 (Central European Standard Time)
db	artefacts
lastAccessTime	Wed Nov 20 2019 18:10:14 GMT+0100 (Central European Standard Time)
name	location

Figure 5: Table metadata

This is achieved through Atlas' classifications, which can group data of the same nature. Moreover, data can be associated with multiple classifications, i.e., be in different data groups at the same time. Figure 10 shows the implementation of the above data organization in Atlas. We associate table *object_origin* with four classifications (i.e. enriched, Artefacts, confidential, 2020). With table *object_origin*

belonging to four classifications, if we want to filter data by maturity, we click on the *enriched* classification to find the table. In sum, classifications allow users to organize data easily and flexibly.

4.2 Data Indexing and Search through thesauri

As mentioned in Section 3.7.2, thesauri are important metadata for project HyperThesau. As an example, we import a thesaurus

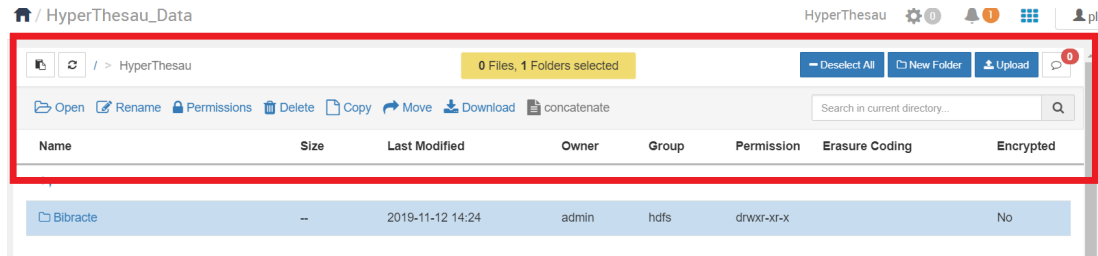


Figure 6: Data upload interface

Key	Value
createTime	Tue Dec 29 2020 00:00:00 GMT+0100 (Central European Standard Time)
fileSize	36763
group	artefacts
isFile	false
isSymlink	false
modifiedTime	Tue Dec 29 2020 00:00:00 GMT+0100 (Central European Standard Time)
name	object-168.txt
numberOfReplicas	0
owner	pliu
path	hdfs://lin02.udl.org:9000/HyperThesau/Artefacts
qualifiedName	hdfs://lin02.udl.org:9000/HyperThesau/Artefacts/object-168.txt

Figure 7: Sample metadata visualization in Altas

provided by the archaeological research facility called *Artefacts* into ArchaeoDAL. This thesaurus is mainly used to index the inventoried archaeological objects of *Artefacts*. Its basic building blocks are *terms* that can be grouped by *categories* (Figure 11).

We can associate any data entity with any term. A data entity can be associated with multiple terms. After we index a data entity with a term, we can search data by using the terms of a thesaurus. For example, we have a database table called *bibliographie* and a file called *204docannexe.csv* that contains information about shields. Suppose we need to associate these two data entities with the term *bouclier* (shield in French). After indexing, we can click on the term *bouclier* to find all data associated with this term (Figure 12).

4.2.1 Data Indexing with Multiple thesauri. One of the biggest challenges of project HyperThesau is that each research facility uses its own thesaurus. Moreover, there is no standard thesaurus. Thus, if we index data with one given thesaurus, archaeologists using another one cannot use ArchaeoDAL. To overcome this challenge,

ArchaeoDAL supports multiple thesauri. Moreover, we can define relations, e.g., synonyms, antonyms, or related terms, between terms of different thesauri. For example, Figure 13 shows a term from a Chinese thesaurus that we set as the synonym of the term *bouclier*. As a result, even though the Chinese term does not relate to any data directly, by using the relations, we can find terms that are linked to actual data. A full video demo of this example can be found online²⁵.

5 CONCLUSION AND FUTURE WORKS

In this article, we first introduce the need of archaeologists for software platforms that can host, process and share new, voluminous and heterogeneous archaeological data. Data lakes looking like a viable solution, we examine different existing data lake solutions and conclude that they are not generic, flexible nor complete enough to fulfill project HyperThesau's requirements.

²⁵<https://youtu.be/OmxsLhk24Xo>

Key	Value
columns (12)	<pre>obj.id obj.id_location obj.id_musee mu.id mu.nom</pre>
createTime	Sat Dec 26 2020 22:26:27 GMT+0100 (Central European Standard Time)
db	artefacts
lastAccessTime	Sat Dec 26 2020 22:26:27 GMT+0100 (Central European Standard Time)
name	objects_origin
owner	pliu
parameters	

Figure 8: Metadata *per se*

Figure 9: Lineage

As a result, we propose a generic, flexible data lake architecture that covers the full data lifecycle. ArchaeoDAL's architecture is

generic because it does not depend on any specific technology. For example, in our current implementation, we use HDFS as the storage

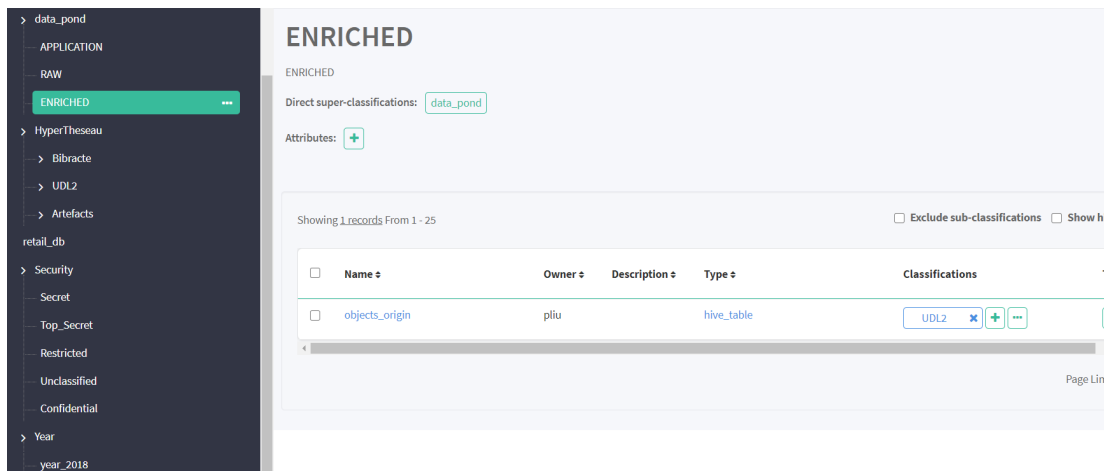


Figure 10: Sample Atlas classification

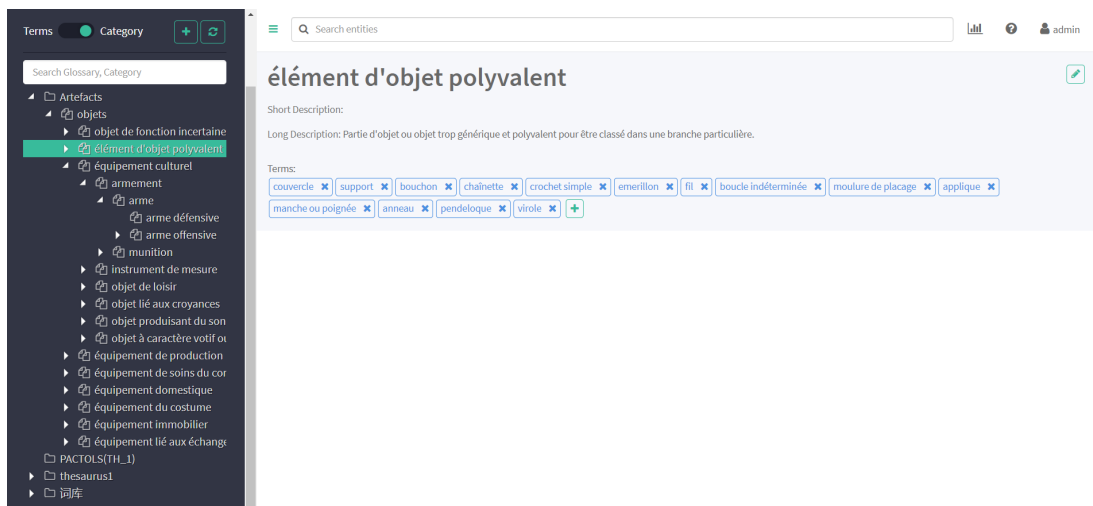


Figure 11: Artefacts' thesaurus in Atlas

layer. Yet, one of our collaborators could easily replace HDFS by Amazon S3. In Section 4.1.4, we demonstrate how to organize data flexibly, which many existing solutions [1, 5, 6, 15] do not allow. In Section 4.1.4, we also demonstrate that ArchaeoDAL can gather metadata automatically during the full data lifecycle. Eventually, many features of ArchaeoDAL are very hard to demonstrate in a paper. Thus, we recorded demo videos that are available online²⁶.

Archaeologists encounter two major problems while working with ArchaeoDAL. First, to associate data and terms in a thesaurus, domain experts are needed. Moreover, this data-terms matching is a very expensive and time-consuming operation. Thus, we plan to use natural language processing techniques to associate data with a thesaurus automatically, calling domain experts only for a *a posteriori* verification.

Second, we handle a lot of images, e.g., aerial photographs and satellite images. It is also very time consuming to detect useful objects in such images. Although some machine learning tasks can already be performed from ArchaeoDAL via Spark-ML, we would like to use deep learning techniques to assist archaeologists in processing images more efficiently.

REFERENCES

- [1] Ian Bird, Simone Campana, Maria Girone, Xavier Espinal, Gavin McCance, and Jaroslava Schovancová. 2019. Architecture and prototype of a WLCG data lake for HL-LHC. *EPJ Web of Conferences* 214 (2019), 04024. <https://doi.org/10.1051/epjconf/201921404024>
- [2] James Dixon. 2010. Pentaho, Hadoop, and Data Lakes. <https://jamesdixon.wordpress.com/2010/10/14/pentaho-hadoop-and-data-lakes>.
- [3] Huang Fang. 2015. Managing data lakes in big data era: What's a data lake and why has it become popular in data management ecosystem. In *CYBER*. IEEE, Shenyang, China, 820–824. <https://doi.org/10.1109/CYBER.2015.7288049>
- [4] Gabriele Gattiglia. 2015. Think big about data: Archaeology and the Big Data challenge. *Archäologische Informationen* 38 (2015). <https://doi.org/10.11588/ai.2015.1.26155>

²⁶https://youtube.com/playlist?list=PLrj4IMV47FypKK5WyEd4Oj3-JnfSuU_H1

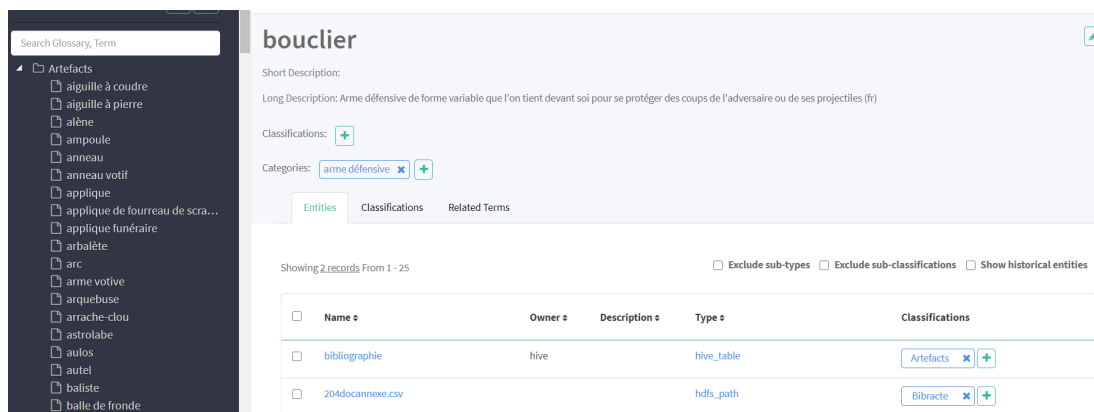


Figure 12: Sample data search through Artefacts' thesaurus

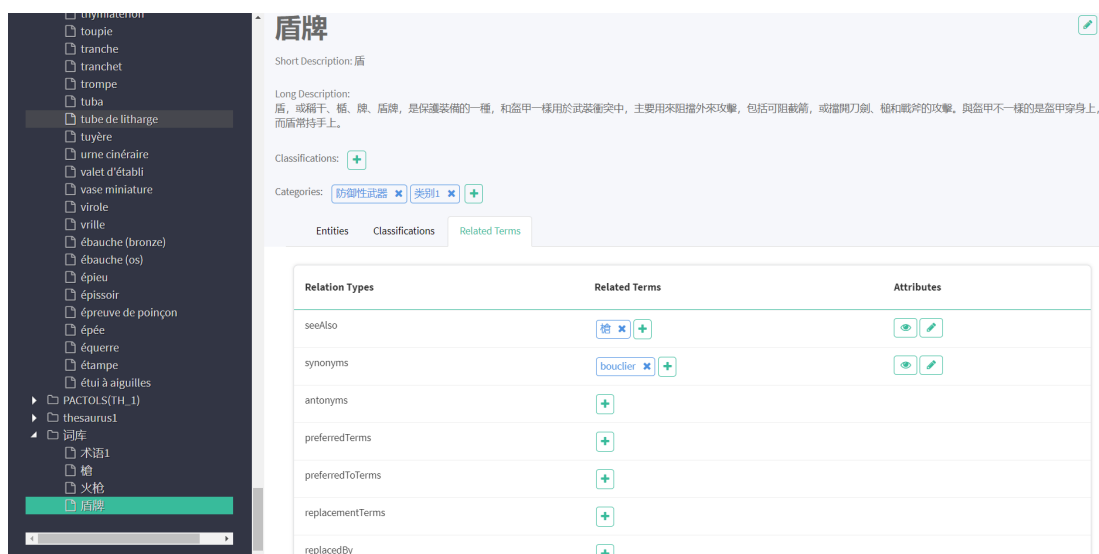


Figure 13: Sample term relations in Artefacts' thesaurus

- [5] Alex Gorelik. 2019. *Architecting the Data Lake*. O'Reilly Media, North Sebastopol, CA, USA, Chapter 7, 133–139.
- [6] Bill Inmon. 2016. *Data Lake Architecture: Designing the Data Lake and Avoiding the Garbage Dump*. Technics Publications, NJ, USA.
- [7] Vijay Khatri and Carol Brown. 2010. Designing data governance. *Commun. ACM* 53 (2010), 148–152. <https://doi.org/10.1145/1629175.1629210>
- [8] Mark McCoy. 2017. Geospatial Big Data and archaeology: Prospects and problems too great to ignore. *Journal of Archaeological Science* 84 (2017), 74–94. <https://doi.org/10.1016/j.jas.2017.06.003>
- [9] Hassan Mehmood, Ekaterina Gilman, Marta Cortes, Panos Kostakos, Andrew Byrne, Katerina Valta, Stavros Tekes, and Jukka Riekk. 2019. Implementing Big Data Lake for Heterogeneous Data Sources. *35th IEEE International Conference on Data Engineering Workshops (ICDEW) 2* (2019), 37–44. <https://doi.org/10.1051/epjconf/201921404024>
- [10] Daniel O'Leary. 2014. Embedding AI and Crowdsourcing in the Big Data Lake. *Intelligent Systems, IEEE* 29 (09 2014), 70–73. <https://doi.org/10.1109/MIS.2014.82>
- [11] Ramakrishna Raju, Rohit Mital, and Daniel Finkelsztin. 2018. Data Lake Architecture for Air Traffic Management. In *2018 IEEE/AIAA 37th Digital Avionics Systems Conference (DASC)*. IEEE, London, UK, 1–6. <https://doi.org/10.1109/DASC.2018.8569361>
- [12] Pegdwendé Nicolas Sawadogo, Etienne Scholly, Cécile Favre, Eric Ferey, Sabine Loudcher, and Jérôme Darmont. 2019. Metadata Systems for Data Lakes: Models and Features. In *1st International Workshop on BI and Big Data Applications (BBIGAP@ADBIS 2019) (Communications in Computer and Information Science, Vol. 1064)*. Springer, Bled, Slovenia, 440–451. <https://doi.org/10.1007/978-3-030-30278-8>
- [13] Etienne Scholly, Pegdwendé Nicolas Sawadogo, Pengfei Liu, Javier Alfonso Espinosa-Oviedo, Cécile Favre, Sabine Loudcher, Jérôme Darmont, and Camille Noël. 2021. Coining goldMEDAL: A New Contribution to Data Lake Generic Metadata Modeling. In *23rd International Workshop on Design, Optimization, Languages and Analytical Processing of Big Data (DOLAP@EDBT/ICDT 2021) (CEUR, Vol. 2840)*. Nicosia, Cyprus, 31–40. <https://hal.archives-ouvertes.fr/hal-03112542>
- [14] John Tomcy and Misra Pankaj. 2017. Lambda Architecture-driven Data Lake. In *Data Lake for Enterprises*. Packt, Chapter 2, 58–77.
- [15] Coral Walker and Hassan H. Alrehamy. 2015. Personal Data Lake with Data Gravity Pull. In *5th IEEE International Conference on Big Data and Cloud Computing (BD-Cloud)*. IEEE, Dalian, China, 160–167. <https://doi.org/10.1109/BDCloud.2015.62>

Towards a continuous forecasting mechanism of parking occupancy in urban environments

Miratul Khusna Mufida

Univ. Polytechnique Hauts-de-France, LAMIH, CNRS,
UMR 8201
Valenciennes, France
Miratul.Mufida@uphf.fr

Thierry Delot

Univ. Polytechnique Hauts-de-France, LAMIH, CNRS,
UMR 8201
Valenciennes, France
Thierry.Delot@uphf.fr

Abdessamad Ait El Cadi

Univ. Polytechnique Hauts-de-France, LAMIH, CNRS,
UMR 8201, INSA Hauts-de-France, F-59313
Valenciennes, France
Abdessamad.AitElCadi@uphf.fr

Martin Trépanier

Polytechnique Montréal, CIRRELT, C.P. 6079, Centre-ville
Montréal, Canada
mtrepanier@polymtl.ca

ABSTRACT

Searching for an available parking space is a stressful and time-consuming task, which leads to increasing traffic and environmental pollution due to the emission of gases. To solve these issues, various solutions relying on information technologies (e.g., wireless networks, sensors, etc.) have been deployed over the last years to help drivers identify available parking spaces. Several recent works have also considered the use of historical data about parking availability and applied learning techniques (e.g., machine learning, deep learning) to estimate the occupancy rates in the near future. In this paper, we not only focus on training forecasting models for different types of parking lots to provide the best accuracy, but also consider the deployment of such a service in real conditions, to solve actual parking occupancy problems. It is therefore needed to continuously provide accurate information to the drivers but also to handle the frequent updates of parking occupancy data. The underlying challenges addressed in the present work so concern (1) the self-tuning of the forecasting model hyper-parameters according to the characteristics of the considered parking lots and (2) the need to maintain the performance of the forecasting model over time.

To demonstrate the effectiveness of our approach, we present in the paper several evaluations using real data provided for different parking lots by the city of Lille in France. The results of these evaluations highlight the accuracy of the forecasts and the ability of our solution to maintain model performance over time.

CCS CONCEPTS

• **Computing methodologies** → **Distributed artificial intelligence**.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](https://permissions.acm.org).

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Association for Computing Machinery.

ACM ISBN 978-1-4503-8991-4/21/07...\$15.00

<https://doi.org/10.1145/3472163.3472265>

KEYWORDS

Smart parking, Machine learning, Continuous forecast, Model optimization, Streaming data, Open data

ACM Reference Format:

Miratul Khusna Mufida, Abdessamad Ait El Cadi, Thierry Delot, and Martin Trépanier. 2021. Towards a continuous forecasting mechanism of parking occupancy in urban environments. In *25th International Database Engineering & Applications Symposium (IDEAS 2021)*, July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3472163.3472265>

1 INTRODUCTION

Searching for an available parking space is stressful, time-consuming and contributes to increasing traffic and pollution. According to [10], this activity is a basic component of urban traffic congestion and generates between 5% and 10% of the traffic in cities and up to 60% in small streets. Moreover, it leads to unnecessary fuel consumption and environment pollution due to the emission of gases. Other studies emphasize the costs of searching for a parking space. For example, a study by the Imperial College in London mentions that, during congested hours, more than 40% of the total fuel consumption is spent while looking for a parking space (Imperial College Urban Energy Systems Project). As a final example, a study by Donald Shoup [23] indicates that, in Westwood Village (a commercial district next to the campus of the University of California, Los Angeles), searching for a parking spot leads every year to about 47,000 gallons of gasoline, 730 tons of CO_2 emissions and 95,000 hours (eleven years) of drivers' time. Last but not least, circling for a vacant parking space may lead to safety issues, as drivers may not pay enough attention to surrounding cyclists and pedestrians at that time.

In order to reduce the time needed to find an available parking space, it may be helpful to provide drivers with accurate and up-to-date information on parking lots occupancy. This information can indeed be collected using various sensors and delivered to drivers through on board-units, smartphones or by exploiting dynamic staking. Moreover, to guide drivers towards locations where they will actually find a place to park, it is crucial to forecast parking occupancy in the near future (typically one hour or less) to inform

drivers while they are still on their way. Recent research works [5, 24] have therefore considered the use of parking history data and compared different techniques (time series, machine learning, etc.). These works show promising results and open new perspectives for parking management.

Thus, in the future, drivers could request a parking close to their destination and/or matching their preferences (walking time to destination, price, etc.) while they are still driving, and be guided towards the best option according to the real-time supply and demand. In this paper, we address the challenges that need to be tackled to reach such an objective, namely (1) provide a forecasting model which easily adapts to the parking lot considered in order to provide the best accuracy and (2) maintain the performance of the forecasting model over time in spite of the frequent changes on parking occupancy data. Therefore, we propose a framework exploiting machine learning techniques, and specially Recurrent Neural Networks - Long Short-Term Memory (RNN-LSTM), to forecast parking occupancy in a short-term future. We show that every parking lot has a different profile in terms of parking occupancy and so requires a dedicated tuning of the forecasting model in order to obtain the best performance. The framework introduced in the paper therefore integrates self-tuning features for different hyper-parameters in order to obtain the best forecast whatever the parking considered. Besides, to maintain a good forecast accuracy, the forecasting models used by our framework have to be continuously updated within a limited time to guarantee a continuous and effective service to drivers.

The structure of the rest of the paper is as follows. In Section 2, we present some related works on building parking occupancy prediction models. Section 3 describes the proposed framework. In section 4, we evaluate this framework and show results that prove the feasibility and the efficiency of our proposal. Finally, we depict our conclusions and draw some prospective lines for future works in Section 5.

2 RELATED WORKS

Recently, managing traffic, mobility, and transport systems in the context of smart cities has attracted the attention of many researchers. Particularly, in the area of parking management, a lot of solutions have been proposed, especially to fulfill the needs of providing sustainable parking services. Solutions cover different parking management aspects and smart parking technology [9, 15]. Nevertheless, these approaches still remain less precise and less efficient due to the lack of real-time forecasts, that adapt to the context of the parking.

Several recent works have focused on forecasting future parking occupancy using various approaches such as Linear Regression (LR), Auto-regressive integrated moving average (ARIMA) [1, 5, 20, 29], Support Vector Regression (SVR), Support Vector Machine (SVM), and Multiple Linear Regression (MLR) [29], Random Forest, Naive Bayes, Markov model [2, 26], Multilayer Perceptron (MLP) [3, 22], Long Short-Term Memory (LSTM) [11, 20, 27] and different other types of neural networks. Most of these works focus on comparing models in order to find the method that provides the best performance [3, 29].

Optimizing the performance of a model by integrating exogenous factors has been considered by [1] in the parking context. In this work, the authors compare Recurrent Neural Network - Long Short Term Memory (RNN-LSTM) and Recurrent Neural Network - Gated Recurrent Units (RNN-GRU) and explain that exogenous factors (e.g., weather, event, time of the day) affect forecasting results. They also show that a model trained with correctly configured hyper-parameters has an effect on the model quality.

By using neural networks, it is possible to develop models integrating spatio-temporal factors such as the Graphical Convolutional Neural Network (GCNN) [27]. In this work, the authors investigate the spatial and temporal relationships of traffic flows in the network, weather, parking meters transactions, network topology, speed and parking lots usage. The prediction horizon considered is 30 minutes and the authors conclude that the forecast is better for business parking lots than for recreational areas among other results.

In [12], the authors use Convolutions Neural Networks (CNN) to extract spatial relations and LSTM to capture temporal correlations. The authors also use a Clustering Augmented Learning Method (CALM) which iterates between clustering and learning to form a robust learning process. An experimentation done using the San Francisco Parking dataset is presented. The results show that CALM outperforms other baseline methods including multi-layer LSTM and Lasso to predict block-level parking occupancy.

Unsurprisingly, existing works focus on obtaining the best model accuracy by applying different approaches. A major technique to improve the model performance consists in finely tuning the model hyper-parameters [16, 19, 25]. The cost of tuning the hyper-parameters by hand is very expensive and requires a lot of expertise. Every machine learning algorithm indeed has a distinct sensitivity concerning hyper-parameters tuning [21]. Therefore, a few works have addressed the automatic tuning of hyper-parameters [13]. The work presented in [13] introduces a data-driven and supervised learning approach applied to hyper-parameters optimization in order to substitute to the expert for tuning the hyper-parameters of the model. The work considers the optimization of a few major hyper-parameters (network selection, number of layers and regularization function) by designing a new algorithm to enhance model performance [13].

Meanwhile, [21] presents several ideas to optimize the selection of hyper-parameters. This work is applied to the management of spatial data in the context of ecological modeling. In [7], model optimization is also considered to implement an evolutionary algorithm for parking occupancy forecasting in urban environments. The authors present real case implementation scenarios. Their approach is accurate and useful for the everyday life.

There are obviously a lot of remaining challenges to tackle to develop robust forecasting model and solve real parking problems. Firstly, the deployment of a tuned/optimized forecasting model is needed. Secondly, it is necessary to maintain the performance of the model over time since updates are continuously produced. Hence, we focus on proposing a solution to tackle these problems. In this paper, we do not focus on exploring the correlations between exogenous factors and parking occupancy, even if some of them are integrated in our forecasting models. We rather extend the work by [7] focusing only on the number of layers and number of neurons

per layer, by considering also hyper-parameters tuning for learning rate, look-back window, batch size, and number of epochs). This self-tuning of hyper-parameters is indeed needed to achieve an efficient model deployment. Besides, an important limitation of existing solutions resides in the lack of integration of real-time information to refresh the forecasting model. This problem is tackled by applying model monitoring and dynamic model selection in [8], but not in the context of parking management.

In the following, we detail our solution to automatically tune hyper-parameters of our forecasting models for parking occupancy. Our objective is to obtain the best performance. Moreover, our forecasting models exploit real-time data as an input which is also exploited to update it and maintain its performances overtime.

3 CONTINUOUS FORECASTING OF PARKING OCCUPANCY

In this paper, we consider the problem of forecasting the occupancy of parking lots in the near future (i.e., up to 1 hour in advance), considering the use of parking historical data. As discussed in section 2, several works have already designed forecasting models and evaluated their effectiveness on parking occupancy data. The authors therefore exploit real data and, after cleaning these data, train models using one subset of the dataset called training data. Then, they compare the accuracy of the forecast using different metrics on an other portion of the data set called testing data. Several models have thus been compared and some of them provide promising results, specially Recurrent Neural Networks - Long Short Term Memory (RNN-LSTM) [1, 5, 22, 27, 28].

The originality of our work is to consider the deployment of forecasting techniques for parking occupancy in real conditions. One possible use-case could for example concern the design of a service continuously providing drivers, during their travel, the estimated future occupancy of the parking lots close to their target destination in order to choose one. In this section, we highlight the challenges to be addressed to design such a service, not only to obtain a good forecast in a given context, but also to provide effective mechanisms to generalize the process to various types of parking lots. Moreover, to propose a continuous forecasting service to drivers, solutions are needed to maintain the quality of the forecast over time. In the following, we first detail the challenges that need to be addressed to achieve these objectives and then present our solutions.

3.1 Challenges

Developing a robust and reliable model to perform accurate forecasts of parking lots occupancy is a challenging task, specially when considering real parking circumstances. In the following, we detail the two main challenges we have identified.

The first challenge to address concerns the design of an efficient model whatever the type of parking lot considered. Indeed, every parking lot has a different "profile" in terms of capacity, surrounding amenities, occupation trend or seasonality. For instance, Figure 1 illustrates the weekly evolution of parking occupancy for two different parking lots in the city of Lille, one close to a commercial center and the other one close to a train station. Thus, a model tuned for a given parking lot may not provide good results if used

on a different one. Moreover, tuning and optimizing a forecasting model implies a lot of different parameters, called hyper-parameters (number of layers, number of neurons per layer, activation function, type of optimizer, loss function, number of epochs, learning rate, the look back windows, batch size, etc.). Numerous values or choices are possible for each one of these parameters and selecting the best combination for all these parameters is so a non-trivial and time-consuming task. Hence, manually tuning the parameters of the forecasting model is not an option and a self-tuning process is required for designing an efficient model for each parking lot where forecasts are needed.

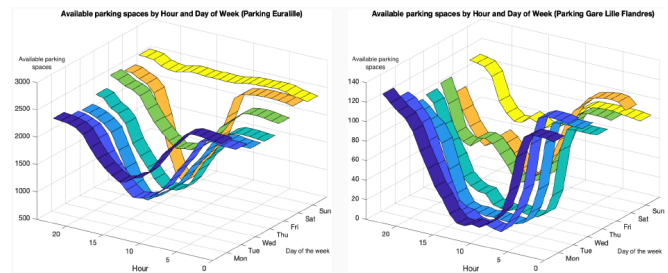


Figure 1: Euralille and Gare Lille Flandres parking profiles

The second challenge to handle when designing a forecasting model concerns the lifetime of this model. The objective when dealing with parking occupancy is obviously to perform a short-term forecast but the question here is how long the forecasting model can effectively operate and provide an accurate forecast. Real-time updates of the parking occupancy are indeed continuously generated and may impact the future forecasts. In this paper, our goal is not only to prove that forecasting the future occupancy of a parking lot can be achieved with a high accuracy but mainly to ensure that the quality of the forecast will remain high as time passes. Hence, data about parking occupancy have to be continuously collected and regularly injected in the forecasting models in order to keep them up-to-date and efficient. This will indeed avoid propagating the error obtained on a forecast done by a model to the following ones, what would cause a very quick degradation of the forecast quality.

In our context, we so have to determine the lifetime of a forecasting model (i.e., the period during which the model gives relevant and significant results). Then, the model has to be updated using data collected since the initial training or since its last update. Let us note that the update phase should be done regularly and within relatively short periods of time to avoid an interruption of the forecast process or a strong degradation of its performance. This update phase will also allow to keep the forecast model fitted to the parking profile. Indeed, these profiles illustrated in Figure 1 may obviously change over time.

3.2 Contributions

In this section, we present our solution to address the challenges introduced in section 3.1, namely the self-tuning of the forecasting model and its retraining over time to keep a good quality forecast.

3.2.1 Self-tuning of the forecasting models. As mentioned in section 2, several works have compared several regression methods (SARIMAX, MLR, SVR) and RNN-LSTM to determine the best one(s) to forecast parking occupancy. These studies concluded that RNN-LSTM provides the best prediction results [1, 5, 22, 27, 28]. Moreover, the experimentation we conducted (see section 4) lead to the same conclusions. Hence, we will focus on RNN-LSTM to forecast parking occupancy in the rest of the paper.

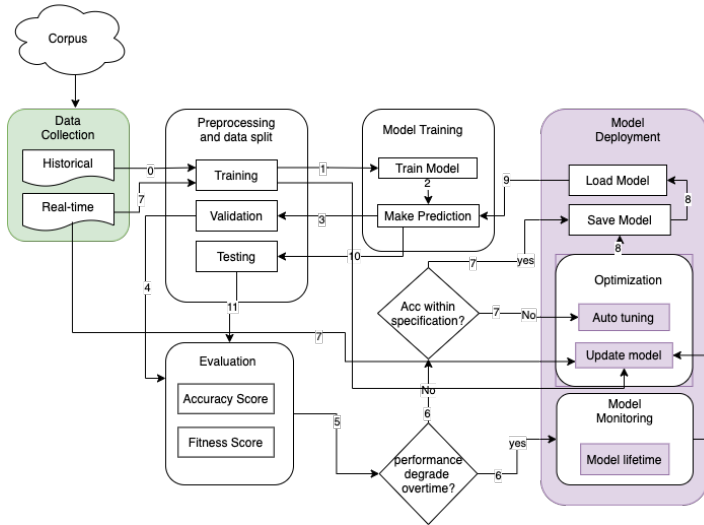


Figure 2: Process Flow

Figure 2 describes the process we propose to manage the heterogeneity of parking profiles and deploy the best forecasting model whatever the characteristics of the target parking lot. It shows the five main steps of our approach: data collection, pre-processing, model training, evaluation and deployment. In this process, a large amount of data about parking occupancy first has to be collected, cleaned and normalized. As usual, this dataset is split in several parts used for training the model in the learning phase and testing it, that is comparing the forecasts obtained with the trained model with the actual data contained in the testing set.

When using neural networks, there are two main customization techniques to optimize the model performance concerning both the architecture of the network (or network selection) and the tuning of the hyper-parameters used during the training phase. This customization phase allows obtaining good performance according to the forecasting objectives. To determine the network structure, five variables have to be considered:

- (1) *Number of layers*: represents the number of hidden layers between the input and output layers in the neural network.
- (2) *Number of neurons*: In the learning process, each neuron at a given layer calculates the weighted sum of inputs, add the bias and execute an activation function.
- (3) *Dropout function*: provides a regularization method used to avoid over-fitting. Over-fitting occurs when a model is constrained to the training set and does not perform well on unseen data.

- (4) *Activation function*: this function is used to add non-linearity to the objective function of the model such as *relu*, *sigmoid* or *tanh*.

Moreover, for hyper-parameters tuning, several items also have to be considered, namely:

- (1) *Type of optimizer*: this item has a crucial role to increase the accuracy of the model. Different types of optimizer can be selected, such as *Stochastic Gradient Descent (SGD)*, *Nesterov Accelerated Gradient*, *Adagrad*, *AdaDelta*, *Adam* or *RMSPROP*.
- (2) *Number of epochs*: this number determines how many times the whole training is executed for the neural network during the training phase.
- (3) *Batch size*: it represents the size of the portions of the dataset used to train the network at each iteration.
- (4) *Loss function*: this function provides a way to calculate the loss, that is a prediction error of the neural network. Again, several choices are possible such as Mean Absolute Error (MAE), Mean Squared Error (MSE), etc.
- (5) *Learning rate*: it determines how much the model should change according to the estimated error each time the model weights are updated.
- (6) *Regularization function and factor*: these elements impose constraints on the weights within the network to avoid over-fitting and thus improve the model performance.

Default values can be determined for these different parameters but obviously, in order to obtain the best model, a tuning phase is necessary. The gap between different combinations on the forecast quality is important and will be illustrated in section 4. This selection process can of course be done manually by changing the number of layers, the number of neurons per layer, the activation function, the learning rate, etc. However, due to the very high number of parameters and possible combinations, this process may be very costly in terms of time and resources. The chances to quickly determine the best combination are low and this process requires a strong expertise for the person in charge of this tuning phase to be effective.

Hence, the big question here is how to make the tuning of our forecast model autonomous in order to best fit with the parking characteristics [13]. Basically, there are two possible strategies to achieve an optimal tuning. First, this can be done by generating an algorithm to automatically tune the hyper-parameters and determine an optimal combination according to the objective. The second one is to develop an algorithm that reduces the set of hyper-parameters to tune by fixing the optimal value for other ones [13]. In the following, we investigate the first approach and present a self-tuning mechanism for the hyper-parameters of our forecasting model. This solution is more generic and can be applied on different parking lots in different contexts.

To simplify the problem, we focus on automatically tuning the possible combinations of hyper-parameters. Thus, auto hyper-parameter tuning applies for several variables and presents the output faster than manual hyper-parameter tuning as we will show in section 4.

Algorithm 1 uses Random Search [6] to determine the optimal value for different parameters of a RNN-LSTM model to forecast parking occupancy in a short-term future. We selected Random Search to tune the parameters because it outperforms grid search [6].

More precisely, this algorithm tries to set the best values for subset of hyper-parameters which are the number of layers, the number of neurons per layer, the learning rate, the look back window, the batch size, type of optimizer and the number of epochs.

Algorithm 1: Random Search for auto-tuning the hyper-parameters

Require: NL Number of layers
Require: NN Number of neurons per layer
Require: LR Learning rate
Require: OP Optimizer
Require: D Dropout
Require: RD Recurrent Dropout
Require: BS Batch Size
Require: NE number of epochs
Require: LW Look back window
Require: AF Activation Function

- 1: $NL \leftarrow [3..10]$
- 2: $LW \leftarrow [2, 4, 6, 12, 24, 48, 96, 192]$
- 3: $NN \leftarrow [2, 4, 8, 16, 32, 64, 128]$
- 4: $AF \leftarrow ['linear', 'ReLU', 'sigmoid', 'tanh', 'SeLu', 'hyperbolic']$
- 5: $D, RD \leftarrow [0.1..1.0]$
- 6: $BS \leftarrow [2, 4, 8, 16, 32, 64, 128]$
- 7: $NE \leftarrow [25, 50, 70, 100]$
- 8: $LR \leftarrow [0.001, 0.003, 0.005, 0.03, 0.05, 0.01, 0.1, 0.3, 0.5]$
- 9: $OP \leftarrow ['ADAM', 'ADAGRAD', 'RMSPROP', 'SGD']$

Ensure: Trained model with particular NL, LW, NN, AF, D, RD, BS, NE, LR and OP
Ensure: Accuracy Metrics
Ensure: Multistep Training and Validation Loss

Create RNN-LSTM Model

- 10: STEP 1 : Set randomly the hyper-parameters (NL, LW, NN, AF, D, RD, BS, NE, LR, OP)
- 11: STEP 2 : $M^* \leftarrow$ Create the model with the chosen hyper-parameters
- 12: STEP 3 : $A^* \leftarrow$ Evaluate the accuracy of the model M^*
- 13: STEP 4 : Pick up another values of hyper-parameters, using a given distribution
- 14: STEP 5 : $M \leftarrow$ Update the model with the new hyper-parameter
- 15: STEP 6 :
- 16: **if** (A is better than A^*) **then**
- 17: $A^* \leftarrow A$ and
- 18: $M^* \leftarrow M$
- 19: **end if**
- 20: STEP 7 :
- 21: **if** (we reach MAX iteration) **then**
- 22: exit
- 23: **return** M^*
- 24: **return** A^*
- 25: **else**
- 26: Go to STEP 4
- 27: **end if**

A few research works already focused on the automation of hyper-parameters tuning in the context of parking management [1] or in different ones, such as supply chain problems [14]. The limitations of these works mainly reside in the limited number of layers and number of neurons per layers they considered [1]. In this paper, we consider the self-tuning of a larger subset of hyper-parameters to improve the model performance.

Once a model is trained, we can thus, using the testing dataset, evaluate its performance. Therefore, we compute different metrics to measure the model performance such as RMSE, MAPE, Adjusted R^2 and tracking signal. We select the best accuracy and so the maximum value for Adjusted R^2 , and the minimum values for RMSE, MAPE and Tracking Signal. Adjusted R^2 is used to determine the explanatory power of our model. It gives the percentage of variation by independent variables that affect the dependent variables. Tracking signal is used to monitor the quality of our prediction model.

3.2.2 Continuous model update. To provide an effective service to drivers and inform them with the future occupancy of close parking lots, the challenge is not only to develop or tune the best forecast model for each parking lot, but also to maintain its performance overtime. The complexity here is to monitor, maintain or improve the model performance over time [8]. The process we propose to continuously maintain a forecasting model efficient over time is depicted in Figure 3.

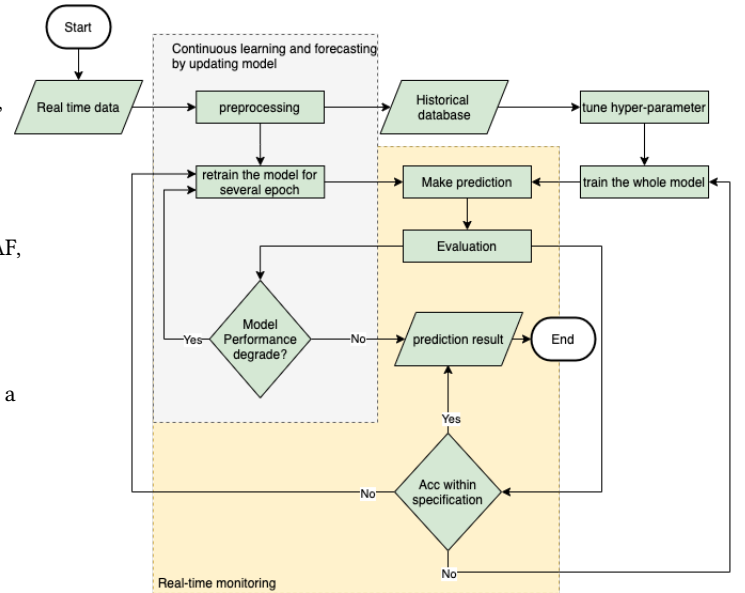


Figure 3: Update process of forecasting models

The first challenge is to determine the moment when an action is needed to update the model. Prediction models indeed use former values or observations to estimate the future ones. When the model starts forecasting values, the first forecasts, obtained with real observations, are so used to determine the next ones. Hence, the potential errors introduced by learning algorithms will accumulate and the quality of the forecasts inevitably decrease over time.

So, once the time when the model has to be updated is determined, the model has to be effectively updated. Obviously, this update has to be performed in a short-time, since it has to be effective before the quality of the forecasts provided by the model decreases too much. Hence, it is not possible to fully retrain the model continuously since this would take too much time. In this work, we rather investigate the use of continuous updates provided by the parking lot to partially retrain the model. We so assume that the parking lots deliver continuous updates of their occupancy every few minutes. Obviously, parking occupancy changes continuously but gathering information every 2 to 5 minutes allow training an efficient model and maintaining its performance over time as we will show in section 4.

The advantage of using neural networks for time series forecasting resides in the possibility to update the model weights when new data is available. Our process to update our forecasting model on the fly is depicted in Figure 3. The yellow block represents the model monitoring mechanism whereas the grey one depicts the update model integrating new data. We start by determining when the model performance starts degrading over time to determine when exactly the model should be updated.

To deploy a forecasting model on a given parking lot, we first set a LSTM model and train it using historical data. We thus obtain a baseline model. This baseline model can provide forecasts during a certain period of time before its performance starts degrading. The baseline model can thus be deployed and evaluated to compare the results with other approaches. Once the baseline model is deployed, we so update it using real-time data continuously collected by our system.

The update mechanism is activated when the model performance degrades. It operates to maintain the model performance over time by injecting real-time data and retrain some epochs. The quality of the model updates depends on the number of additional epochs performed on new data to update the weights of the model. Obviously, the size of the dataset used to retrain the model will also impact the time needed to perform the update. Experimental results will be presented in Section 4 to show the effectiveness of the update and the impact of the number of epochs performed.

The key points of this process reside in the availability of real-time data and the monitoring procedure to know when the model needs to be updated. Our mechanism so relies on a real-time monitoring and update mechanism to achieve continuous learning and forecasting. The real-time monitoring mechanism determines when the model should be updated by detecting that performance starts degrading. Then, retraining the model regularly with new unseen data allows maintaining its performance over time and supporting possible changes in the occupancy of the parking lot.

In this section, we have presented the main challenges to address as well as our solutions to provide drivers with parking occupancy forecasts in a continuous manner. In the following one, we depict the different evaluations realized to show the performance of our self-tuning mechanism, to determine the basic lifetime of the forecasting models produced and maintain their performance over time.

4 EXPERIMENTAL STUDY AND RESULT

In this section, we describe the experiments realized to show the efficiency of our continuous learning mechanism. These experiments have been conducted using the applied machine learning frameworks TensorFlow and Keras.

4.1 Dataset description and basic settings

Real-parking data provided by Lille European Metropolis (MEL), the largest city in the North of France including 1.2 million people, have been used to build and evaluate our models. The dataset contains data about the occupancy of 27 parking lots, representing a total of 18,180 parking spaces, from December 2018 to February 2020. The number of cars parked is refreshed every few minutes and the updates can be easily accessed online¹. Each parking area has a different capacity and occupancy "profile" as illustrated in Figure 4 for Parking Euralille and Figure 5 for parking Gare Lille Europe. Parking Euralille, is located near a shopping mall whereas parking Gare Lille Europe is next to a train station.

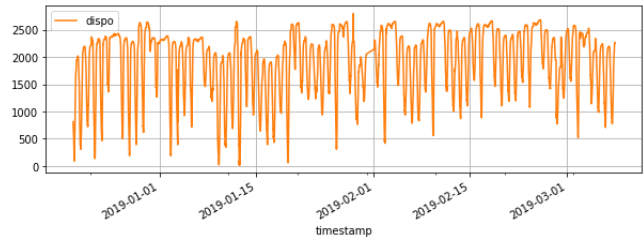


Figure 4: Occupancy of parking Euralille over time

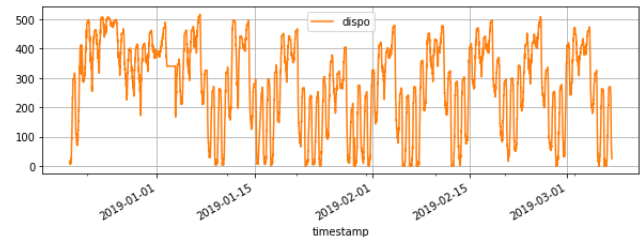


Figure 5: Occupancy of parking Gare Lille Europe over time

The raw data are stored in daily csv files. Each file contains the daily data for all parking areas. It consists of 12 columns namely: label (parking name), address, city, status (open or closed), number of available spaces, maximum capacity, date, parking id, coordinates, geometry, display panels, timestamp.

In the following, we evaluate our contributions to continuously forecast parking occupancy using this real dataset. However, the raw data that we use to deploy the model are noisy and some values are missing. In order to make the dataset compliant with the training step, we first perform data cleaning. Thus, to handle missing values in our time series, we decided to replace them by the value observed at the same time during the preceding week. The different

¹For more information, see <https://opendata.lillemetropole.fr/explore/dataset/disponibilite-parkings/information/>.

phases of the model development can then start: training, validation, optimization with (auto) hyper-parameter tuning, testing and evaluation. For the experiments considered in the rest of this section, we used a portion of the dataset ranging from December 19th, 2018 to March 7th, 2019. 80% of the data were used as training data, 10% for validation and 10% for testing. To evaluate the quality of the models generated and select the best one, several metrics are considered: Accuracy, root mean squared error (RMSE), Mean Absolute Percentage Error (MAPE), Adjusted R^2 and Tracking Signal. MAPE is used to present the forecast error as a percentage, RMSE to present the gap between the real data and the forecast using the commonly used loss function in the related works. Adjusted R^2 allows determining the percentage of variation by independent variables that affect the dependent variables. The closer to 1 the better it is. Finally, tracking signal is used to monitor and identify the prediction deviation.

4.2 Results

In this section, we propose an evaluation of our different contributions. We start by illustrating how we deploy a baseline model and by comparing the performance provided by RNN-LSTM to the other forecasting techniques.

4.2.1 Models Performance Comparison. Our objective here is to justify why we focused on RNN-LSTM in section 3. We therefore compared the performance of different forecasting models such as SARIMAX, MLR, SVR and RNN-LSTM to perform short-term predictions on time series dataset. To know which model performs the best, we compared the forecasts provided by the four models listed above using the Euralille dataset. After training the models, we evaluate their performance and provide the MAE, MSE, MAPE and RMSE errors in Table 1, as well as, the tracking signal in Figure 6. In Table 1 and Figure 6, the smallest error indicates the best model. Also, by comparing the values obtained for the training sets to the ones collected with the testing sets, the gap indicates the quality of the fitting of the model: does it over-fit, under-fit, or does it have the best fit. The best fit model performs well to forecast unseen data whereas over fitting or under-fitting models have problem to perform well on different datasets.

Table 1: Errors comparison of four different models

Loss Function	RNN-LSTM	SARIMAX	MLR	SVR
MAE Training	0.13	0.28	0.12	0.27
MAE Testing	0.13	0.11	0.18	0.11
MSE Training	0.03	0.12	0.31	0.13
MSE Testing	0.13	0.46	0.47	0.26
MAPE Training	17.05%	39.71%	17.02%	38.29%
MAPE Testing	18.43%	15.60%	25.53%	15.60 %
RMSE Training	0.17	0.34	0.55	0.36
RMSE Testing	0.36	0.68	0.69	0.51

The error determined using the training set is known as model bias. The common term for the testing set is variance. We here aim at obtaining the minimum value for both bias and variance. Tracking signal is used to represent the forecasting bias of the model. MAE,

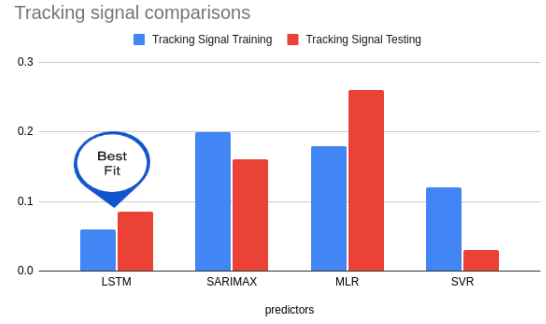


Figure 6: Tracking signal comparison for 4 different predictors

MSE, RMSE, and MAPE are used to exhibit the forecasting model's variation. Based on the results presented in Table 1 and Figure 6, we observe that RNN-LSTM outperforms SARIMAX, MLR and SVR with the minimum error and a best fit. Hence, we will focus in the rest of this section on RNN-LSTM.

The first result using a single parking dataset (Euralille) shows that RNN-LSTM outperforms the other models. In section 4.2.2, we will study the possibility to use a model trained with a particular dataset to performs forecasts for another (previously unseen) dataset.

4.2.2 Generalization of model deployment to different parking areas. Our objective is to forecast for a large number of parking lots (i.e., all the parking lots in a city for instance). An option here could be to build a general model (super model) using all the data available and then use it to forecast the occupancy of all considered parking lots. This is a challenging task [1, 7] and this does not work properly due to the specificity of each parking lot in terms of occupancy trend. So, we rather decided to optimize one model per target parking lot. To illustrate our point, we present the results obtained by training and optimizing a model using a single parking area (i.e., Euralille) – Table 2 shows the list of hyper-parameters selected for the model after the tuning phase – Then, we applied the same optimized model on another parking area (i.e., Gare Lille Europe). The forecasting results on the original parking – used for the tuning – have scores of 0.66% for MAPE and 0.126 for RMSE, while on the new one, we have, respectively, the scores of 17,6% and 0.3753. The quality of the forecast significantly decreases as shown in Table 3. In conclusion, to have the best performance, we have to tune the hyper-parameters of each parking lot, by considering each parking profile. A unique forecasting model can indeed not provide good forecasts, whatever the parking lot. In subsection 4.2.3, we focus on hyper-parameters tuning.

4.2.3 Auto-tuning of hyper-parameters. To actually deploy the best model for each parking lot, it is crucial to automatically determine the best hyper-parameters. Tuning hyper-parameters is an effective technique to increase model performance, as shown in Table 4. This approach is obviously time-consuming due to the huge search space (i.e., number of parameters and number of combinations). Its duration also depends on the selected number of epochs and on the performance of the computer used, but training a single model

Table 2: Values of hyper-parameters selected for Euralille

Hyper-parameter	Tuning
Batch size	32
Epoch	50
Prediction horizon	12
Look back window	2016
Step	12
Activation Function	RMSPROP
Number of layer	3
Neuron per layer	128/64/12
Loss function	MAE, Accuracy

Table 3: Comparison of errors when using model trained on one parking lot (Euralille) on different parking lots

Parking	RMSE	MAPE
Euralille	0.1260	0.66%
Gare Lille Europe	0.3753	17.6%

**Figure 7: MAPE evolution over time****Table 4: Impact of hyper-parameters tuning on model performance**

Model	RMSE	MAPE
Baseline	0.37	41.29%
Tuning 1	0.28	21.86%
Tuning 2	0.29	30.57%
Tuning 3	0.28	30.6%
Tuning 4	0.0.22	28.21%
Tuning 5	0.13	21.64%

usually takes several tens of minutes (57 minutes for one training in our experiments).

To obtain the best fit model that is able to forecast unseen data in the future, auto-tuning the hyper-parameters provides the best solution. From our experience, using the default parameters for tuning a model leads to problems of over-fitting/ under-fitting as

shown in Figure 8. Avoiding over-fitting and under-fitting is an important objective. Over-fitting indeed refers to the ability of the model to fit well to the training data but not to generalize to another dataset. Under-fitting corresponds to the characteristic of a model both unable to fit to the training set, neither to another dataset. Another possibility is to choose the combination of hyper-parameters by hand. This is also a challenging task considering the size of the search space. Also, since we have to deal with various profiles of parking lots, this would require an expert to tune each model before it can be exploited. Hence, auto-tuning constitutes the best optimization technique and the easiest way to find the best fit model setting. It indeed finds the training and validation loss that converge at one point, as illustrated in Figure 9, contrary to the default settings shown in Figure 8. Table 5 and Table 6 show that the auto-tuning of hyper-parameters is efficient and effective to obtain the best model performance.

Table 5 shows the hyper-parameters used for the default tuning (for both parking lots) and the ones selected by algorithm 1 introduced in section 3.2 for parking Euralille and GLE. Table 6 shows the improvement of the performance resulting from the auto-tuning phase with a strong reduction of the error for both parking lots.

The auto-tuning of hyper-parameters facilitates model deployment by automatically selecting the best hyper-parameters for the training phase. Another challenge is related to the fact that the predictor performance of each model declines over time. Subsection 4.2.4 thus focuses on the necessary updates of the forecasting models to avoid a degradation of the forecast quality over time.

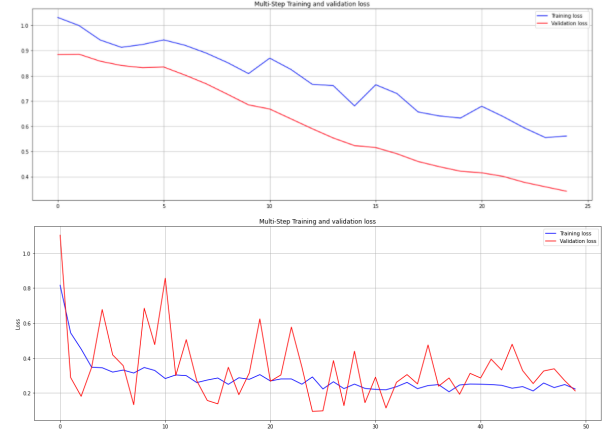
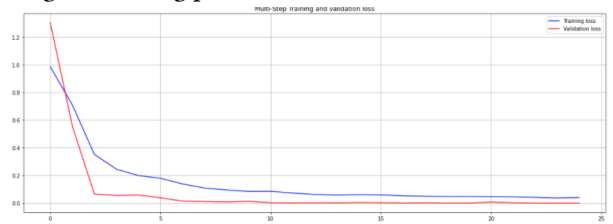
**Figure 8: Manual hyper-parameters tuning and under-fitting/over-fitting problems****Figure 9: Impact of hyper-parameters auto-tuning on under-fitting/over-fitting problems**

Table 5: List of default and optimal hyper-parameters after tuning for Euralille and Gare Lille Europe

Hyper-parameters	Default Euralille	Default GLE	Euralille	GLE
Batch size	1	1	64	32
Epoch	10	10	25/50/70/100	35/40/50
Prediction horizon	72	72	2/12/48/72	3/12
Look back window	2016	2016	1800/2016	2008
Step	12	12	12	12
Activation Function	ADAM	ADAM	RMSPROP	ADAM
Number of layer	3	3	3	5
Neuron per layer	16/32/72	16/32/72	128/64/12	4/128/64/16/128
Loss function	MAE	MAE	MAE	MAE
RMSE	0.3647	0.8751	0.1260	0.5569
MAPE	18.43%	35.85	0.66%	0.03%

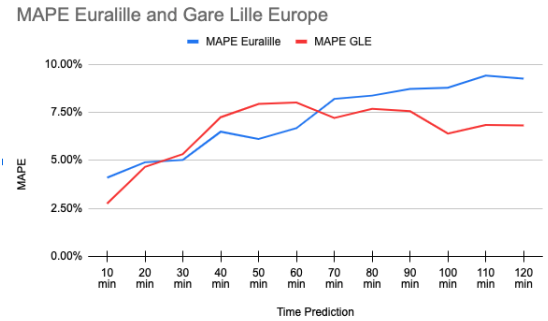
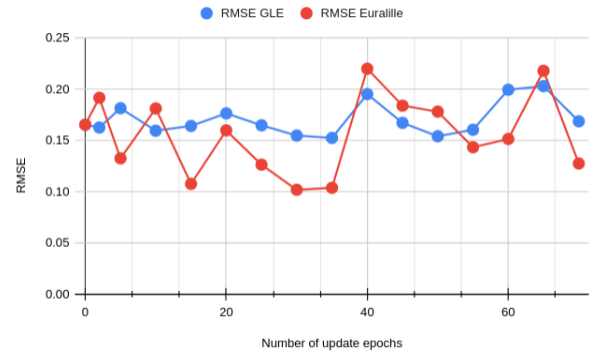
Table 6: Forecasting performance improvement (RMSE and MAPE) after auto-tuning of the hyper-parameters

Model	RMSE	MAPE
Default Tuning Euralille	0.3647	18.43%
Default Tuning GLE	0.8751	35.85%
Self Tuning Euralille	0.1260	0.66%
Self Tuning GLE	0.5569	0.03%

4.2.4 Maintaining model performance over time. In this section, we evaluate how long the forecast provided by a model, once tuned, is accurate enough to be used in a real situation. We so focus on a single day and observe the evolution of the forecast quality over time. Figure 10 illustrates the evolution of the MAPE for forecasts performed every 10 minutes on two different parking lots (Euralille and GLE) during 120 minutes. The MAPE captures the evolution of the error (i.e., the difference between the forecast and the observed value) over time. We observe in Figure 7 that the model performance degrades over time. The error thus exceeds the threshold of 5% after 30 minutes. This information is very useful to determine when the model needs to be retrained to keep a good forecast quality.

To avoid that the model performance degrades too much and that MAPE score exceeds a particular threshold (e.g., 5%), we partly retrain the model by running several epochs and injecting the last updates of the parking occupancy. Our previous experiments on both parking areas also indicate that the model performs well during a relatively short period of time (between 10 and 30 minutes) shown in figure 10. Hence, the retraining phase should be very quick and a full retraining of the model is not possible due to a lack of time. Therefore, we investigated the best or lowest number of epochs needed to improve the performance of an already tuned model. Figure 11 shows the evolution of RMSE for both parking lots according to the number of epochs considered when retraining the model. The optimal retraining is obtained for both parking lots between 30 and 35 epochs. The time needed for this retraining phase ranges between 5 and 6 minutes.

Finally, Figure 12 illustrates the effectiveness of our update mechanism. In this Figure, we indeed see the evolution of the MAPE

**Figure 10: MAPE evolution over time for parking Euralille and Gare Lille Europe****Figure 11: RMSE evolution according to the number of epochs for parking Euralille and Gare Lille Europe**

error over time passed since the training of the model while applying our update process every 30 minutes. Contrary to the scenario considered in Figure 10, we observe that the MAPE error is then maintained under the threshold of 5%. This result shows that our update model mechanism improves the model performance by continuously injecting real data.

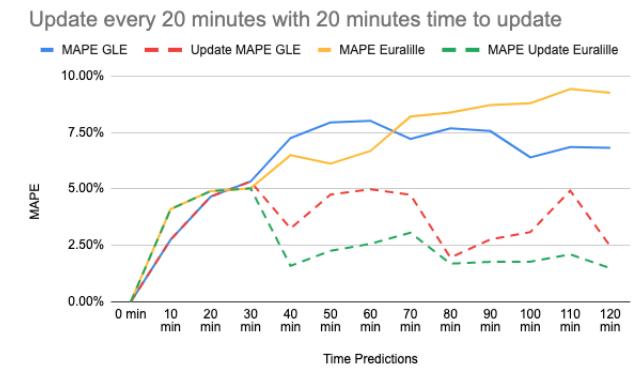


Figure 12: MAPE evolution over time (with updates) for parking Euralille and Gare Lille Europe

5 CONCLUSION AND FUTURE WORK

In this paper, we have presented solutions to continuously forecast parking availability using both historical data and real-time updates. Therefore, we have proposed a solution that can automatically adjust the forecasting model to each parking lot, as well as a mechanism to keep the model efficient over time.

Besides, parking is competitive by nature because after making a choice to visit a particular slot, the success in obtaining that slot will depend on the choice of other closer vehicles [4]. Obviously, forecasting parking availability in the near future does not solve this issue. However, features to provide real-time forecast available parking space may significantly improve frameworks designed to allocate vehicles to available parking spaces [17, 18], and thus manage competition, by providing them information about the future parking occupancy to make better decisions.

6 ACKNOWLEDGEMENTS

This work was supported by Univ. Polytechnics Hauts de France, CNRS UMR 8201, Valenciennes, France, CIRRELT, Montreal, Canada and State Polytechnics of Batam Indonesia and Funded by the Indonesian Minister of Education and Culture.

REFERENCES

- [1] Jamie Arjona, M^aPaz Linares, Josep Casanovas-Garcia, and Juan José Vázquez. Improving parking availability information using deep learning techniques. *Transportation Research Procedia*, 47:385–392, 2020.
- [2] Yacine Atif, Sogol Kharrazi, Ding Jianguo, and Sten F Andler. Internet of things data analytics for parking availability prediction and guidance. *Transactions on Emerging Telecommunications Technologies*, 31(5):e3862, 2020.
- [3] Faraz Malik Awan, Yasir Saleem, Roberto Minerva, and Noel Crespi. A comparative analysis of machine/deep learning models for parking space availability prediction. *Sensors*, 20(1):322, 2020.
- [4] Daniel Ayala, Ouri Wolfson, Bo Xu, Bhaskar DasGupta, and Jie Lin. Parking in competitive settings: A gravitational approach. In *2012 IEEE 13th International Conference on Mobile Data Management*, pages 27–32. IEEE, 2012.
- [5] Claudio Badii, Paolo Nesi, and Irene Paoli. Predicting available parking slots on critical and regular services by exploiting a range of open data. *IEEE Access*, 6:44059–44071, 2018.
- [6] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *The Journal of Machine Learning Research*, 13(1):281–305, 2012.
- [7] Andrés Camero, Jamal Toutouh, Daniel H Stolfi, and Enrique Alba. Evolutionary deep learning for car park occupancy prediction in smart cities. In *International Conference on Learning and Intelligent Optimization*, pages 386–401. Springer, 2018.
- [8] Rosa Candela, Pietro Michiardi, Maurizio Filippone, and Maria A Zuluaga. Model monitoring and dynamic model selection in travel time-series forecasting. *arXiv preprint arXiv:2003.07268*, 2020.
- [9] Thierry Delot, Sergio Ilarri, Sylvain Lecomte, and Nicolas Cenerario. Sharing with caution: Managing parking spaces in vehicular networks. *Mobile Information Systems*, 9(1):69–98, 2013.
- [10] Eric Gantelet and Amélie Lefauconnier. The time looking for a parking space: strategies, associated nuisances and stakes of parking management in France. In *European Transport Conference*, 2006.
- [11] Felix A Gers, Douglas Eck, and Jürgen Schmidhuber. Applying lstm to time series predictable through time-window approaches. In *Neural Nets WIRN Vietri-01*, pages 193–200. Springer, 2002.
- [12] Soumya Suvra Ghosal, Abderrahman Bani, Amine Amrouss, and Issmail El Hallaoui. A deep learning approach to predict parking occupancy using cluster augmented learning method. In *2019 International Conference on Data Mining Workshops (ICDMW)*, pages 581–586, 2019.
- [13] Frank Hutter, Jörg Lücke, and Lars Schmidt-Thieme. Beyond manual tuning of hyperparameters. *KI-Künstliche Intelligenz*, 29(4):329–337, 2015.
- [14] Anirut Kantasa-ard, Maroua Nouiri, Abdelghani Bekrar, Abdessamad Ait el Cadi, and Yves Sallez. Machine learning for demand forecasting in the physical internet: a case study of agricultural products in thailand. *International Journal of Production Research*, pages 1–25, 2020.
- [15] Trista Lin, Herve Rivano, and Frederic Le Mouel. A survey of smart parking solutions. *IEEE Transactions on Intelligent Transportation Systems*, 18(12):3229–3253, 2017.
- [16] Jonathan Lorraine and David Duvenaud. Stochastic hyperparameter optimization through hypernetworks. *CoRR*, abs/1802.09419, 2018.
- [17] Marko Mladenović, Thierry Delot, Gilbert Laporte, and Christophe Wilbaut. The parking allocation problem for connected vehicles. *Journal of Heuristics*, 26(3):377–399, 2020.
- [18] Marko Mladenović, Thierry Delot, Gilbert Laporte, and Christophe Wilbaut. A scalable dynamic parking allocation framework. *Computers & Operations Research*, 125:105080, 2021.
- [19] Steven J Nowlan and Geoffrey E Hinton. Simplifying neural networks by soft weight-sharing. *Neural computation*, 4(4):473–493, 1992.
- [20] Alaa Sagheer and Mostafa Kotb. Time series forecasting of petroleum production using deep lstm recurrent networks. *Neurocomputing*, 323:203–213, 2019.
- [21] Patrick Schratz, Jannes Muenchow, Eugenia Iturritxa, Jakob Richter, and Alexander Brenning. Hyperparameter tuning and performance assessment of statistical and machine-learning algorithms using spatial data. *Ecological Modelling*, 406:109–120, 2019.
- [22] Wei Shao, Yu Zhang, Bin Guo, Kai Qin, Jeffrey Chan, and Flora D Salim. Parking availability prediction with long short term memory model. In *International Conference on Green, Pervasive, and Cloud Computing*, pages 124–137. Springer, 2018.
- [23] Donald Shoup. Cruising for parking. *Access Magazine*, 1(30):16–23, 2007.
- [24] Daniel H. Stolfi, Enrique Alba, and Xin Yao. Can i park in the city center? predicting car park occupancy rates in smart cities. *Journal of Urban Technology*, 0(0):1–15, 2019.
- [25] Vladimir Vapnik, Esther Levin, and Yann Le Cun. Measuring the vc-dimension of a learning machine. *Neural computation*, 6(5):851–876, 1994.
- [26] Jun Xiao, Yingyan Lou, and Joshua Frisby. How likely am i to find parking?—a practical model-based framework for predicting parking availability. *Transportation Research Part B: Methodological*, 112:19–39, 2018.
- [27] Shuguan Yang, Wei Ma, Xidong Pi, and Sean Qian. A deep learning approach to real-time parking occupancy prediction in transportation networks incorporating multiple spatio-temporal data sources. *Transportation Research Part C: Emerging Technologies*, 107:248 – 265, 2019.
- [28] Yunjie Zhao, Gavril Giurgiu, Raul Cajias, Leon Stenneth, and Eric Linder. Method and apparatus for building a parking occupancy model, December 6 2018. US Patent App. 15/610,237.
- [29] Ziyao Zhao and Yi Zhang. A comparative study of parking occupancy prediction methods considering parking type and parking scale. *Journal of Advanced Transportation*, 2020, 2020.

Predicting late symptoms of head and neck cancer treatment using LSTM and patient reported outcomes

Yaohua Wang
Electrical and Computer Engineering
University of Iowa

Lisanne Van Dijk
Radiation Oncology
UT M.D. Anderson Cancer Center

Abdallah S. R. Mohamed
Radiation Oncology
UT M.D. Anderson Cancer Center

Clifton David Fuller
Radiation Oncology
UT M.D. Anderson Cancer Center

Xinhua Zhang
Computer Science
University of Illinois at Chicago

G. Elisabeta Marai
Computer Science
University of Illinois at Chicago

Guadalupe Canahuat
Electrical and Computer Engineering
University of Iowa

Abstract

Patient-Reported Outcome (PRO) surveys are used to monitor patients' symptoms during and after cancer treatment. Late symptoms refer to those experienced after treatment. While most patients experience severe symptoms during treatment, these usually subside in the late stage. However, for some patients, late toxicities persist negatively affecting the patient's quality of life (QoL). In the case of head and neck cancer patients, PRO surveys are recorded every week during the patient's visit to the clinic and at different follow-up times after the treatment has concluded. In this paper, we model the PRO data as a time-series and apply Long-Short Term Memory (LSTM) neural networks for predicting symptom severity in the late stage. The PRO data used in this project corresponds to MD Anderson Symptom Inventory (MDASI) questionnaires collected from head and neck cancer patients treated at the MD Anderson Cancer Center. We show that the LSTM model is effective in predicting symptom ratings under the RMSE and NRMSE metrics. Our experiments show that the LSTM model also outperforms other machine learning models and time-series prediction models for these data.

CCS Concepts

• Computing methodologies → Neural networks.

Keywords

Long Short-Term Memory (LSTM), Patient Reported Outcomes (PRO), Late Toxicity, Symptom Severity Prediction

ACM Reference Format:

Yaohua Wang, Lisanne Van Dijk, Abdallah S. R. Mohamed, Clifton David Fuller, Xinhua Zhang, G. Elisabeta Marai, and Guadalupe Canahuat. 2021. Predicting late symptoms of head and neck cancer treatment using LSTM and patient reported outcomes. In *25th International Database Engineering & Applications Symposium (IDEAS 2021)*, July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 6 pages. <https://doi.org/10.1145/3472163.3472177>

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Copyright held by the owner/author(s).

ACM ISBN 978-1-4503-8991-4/21/07.

<https://doi.org/10.1145/3472163.3472177>

1 Introduction

During head and neck cancer treatment, patients may experience different symptoms with different severity during and after treatment [4, 27, 28]. A commonly used way to monitor patients' symptoms is to record symptom severity or occurrence through a questionnaire survey, which is commonly known as Patient-Reported Outcomes (PRO) data. Much research is done over these PRO data in order to identify symptoms in an early stage and guide treatment decisions, as well as investigating the relationships among these symptoms [19, 24]. In this work, we use data collected from head and neck cancer patients treated at the M.D. Anderson Cancer Center using the M.D. Anderson Symptom Inventory questionnaire [5] and more specifically, the Head-Neck Module (MDASI-HN) [22]. The module is comprised of 28 questions, 13 referring to core symptoms related to cancer (systemic), 9 to head and neck symptoms (local), and the remaining 6 to symptom-burden interference with daily activities (life general). Patients rated the severity of their symptoms on a scale of 0 to 10 with 0 being mild or no existence and 10 being very severe (the worst imaginable). During treatment, symptoms are experienced with greater severity than after treatment. Ideally, we would like to see that all symptoms have receded in the late stage (e.g. a year after treatment), but in some cases, symptoms persist affecting the Quality-of-Life (QoL) of the patients in the long term.

Previous research over the MDASI-HN PRO data applies factor analysis and cluster analysis to cluster and investigate symptom progression [25]. These researches look at a particular snapshot in time to cluster either the patients, by their experienced symptoms, or the symptoms given the patient's ratings. In this work, we approached the problem from a different perspective by modeling the PRO data as a time-series and applying the Long-Short Term Memory (LSTM) Neural Network model [12] to predict late symptom's rating 6 weeks and 12 months after treatment. Since PRO data is self-reported, patients may skip questions or entire questionnaires altogether, resulting in many missing values. To overcome this issue, we applied several methods for missing data imputation and evaluate the performance of the LSTM model for each method. We show that the LSTM model is effective in predicting symptom ratings under the RMSE and NRMSE metrics. In our experiments, the LSTM model also outperforms other machine learning models. Furthermore, we show that for this particular task as it has been observed in other

domains, the more data used, the better the model obtained, even when the data needs to be imputed.

The main contributions of this paper can be summarized as follows. This is the first work that looks at predicting late toxicity for head and neck cancer patients using LSTM. We evaluate different imputation methods for completing the data, including applying LSTM recursively. We compare the LSTM performance against ARIMA and other machine learning models.

2 Related Work

MDASI-HN PRO Data. PRO data has been widely collected physically and electronically in the clinical area since it has an important meaning of evaluating the treatment benefits [6]. The PRO data used in this project is an MDASI-HN [22] questionnaire with 28 symptoms to be rated on a scale from 0 to 10 with 0 being mild or no symptoms and 10 being very severe, during and after treatment. As shown in Table 1, the 28 symptoms can be divided into three types of toxicity. All patients are asked to fill MDASI-HN surveys before the start of treatment (baseline) and then weekly for the 6 weeks of the duration of the treatment. Patients are also asked at their follow-up visits 6 weeks, 6 months, and 12 months after treatment.

Using the MDASI-HN PRO data, several studies focus on identifying symptom clusters at a single timepoint [10, 14, 23]. Prior research mainly used two methods to find symptom clusters, one is factor analysis such as principal component analysis and the other one is cluster analysis such as hierarchical agglomerative clustering [2, 7, 11, 25]. These studies focus on a single time point analysis, whereas we model the PRO data as a time series.

Time Series Prediction and Imputation. Prediction of time series is typically done by looking at the previous values in the series and deciding the value at the current time step. Auto-regressive Integrated Moving Average (ARIMA) [1] is a commonly used method for the prediction of time series. The model combines the Auto-regressive (AR) and Moving Average (MA) models that are suitable for univariate time series modeling. In the AR model, the output depends on its lags while in the MA model, the output depends only on the lagged forecast errors. More recently, Long short term memory (LSTM) Recurrent Neural Networks have gained more popularity in time series prediction [9] and healthcare domain [15]. Specifically, LSTM networks were used to mimic the pathologist decision and other diagnostic applications [18, 30]; LSTM networks were used to recognize sleep patterns in multi-variate time-series clinical measurements [16]. However, to the best of our knowledge, it has not been previously applied to PRO data.

Data imputation methods such as Multiple Imputation by Chained Equations (MICE) [3, 21], linear regression, Kalman filtering [13], among others, can be used to impute time series data.

3 Proposed Approach

In this section, we first describe the methodological approach including data pre-processing and the methods used for data imputation.

3.1 Long Short Term Memory (LSTM)

Since PRO data with patients self-reporting on the severity of their symptoms is collected over time, we model it as a time series. If we can learn from the patients' answers over a period of time and

Toxicity	Symptoms
Systemic	fatigue, constipation, nausea, sleep, memory, appetite, drowsy, vomit, numb
Local	pain, mucus, swallow, choke, voice, skin, taste, mucositis, teeth, shortness of breath (SOB), dry mouth
Life general	general activity, mood, work, relations, walking, enjoy, distress, sad

Table 1: The 28 symptoms in the MDASI-HN questionnaire grouped into three types of toxicities.

predict their answers next week or in 6 months, we could proactively make recommendations to minimize the symptom's burden and therefore, improve the patients' quality of life. For example, we can record patients' responses to different symptoms from week 0 to week 5 during the treatment and predict what the ratings for those symptoms would be in week 6. Such prediction, if accurate, can be useful to make patients aware of the risks and prescribe exercises or medication that can help patients cope with the symptoms to avoid having to adjust treatment and improve the long-term quality of life of the patients.

Long short term memory (LSTM) neural networks are a type of recurrent neural network (RNN) proven effective in predicting time series [12]. Unlike a traditional neural network, LSTMs have a feedback structure to store the memory of the events happened in the past and use it as a parameter in prediction. The basic structure of an LSTM model takes 3 different pieces of information: the current input data, the short-term memory (hidden states) from the previous cell, and the long-term memory (cell state). This 3-dimensional data structure (number of samples, number of time steps, number of parallel time series on features) is pushed through the LSTM gates, which are used to regulate the information to be kept or discarded, i.e. selectively remove any irrelevant information. The LSTM model is able to memorize the time-series pattern of each patient's response and be able to predict late toxicity. Another advantage of the LSTM is the diversity of the inputs and outputs. LSTM can handle multiple predictions simultaneously. In a many-to-one mode, the LSTM would learn from many patients and predict one symptom. In a many-to-many mode, the LSTM would learn from many patients and predict all 28 symptoms for the test data. Taking advantage of this, we are able to generate predictions for all 28 symptoms using one trained LSTM with many-to-many mode.

To feed the data into the LSTM model, the original data was transformed into a 3-dimensional array where the first dimension corresponds to the patients, the second dimension to the time steps, and the third dimension to the symptoms. The number of patients corresponds to the number of samples in the training data. The number of time steps depends on what late toxicity we are evaluating. For the toxicity at 6 weeks after treatment, the number of time steps is 6, while for the toxicity at 12 months after treatment, the number of time steps is 11. The number of symptoms is always 28.

3.2 Data Imputation

Many of the symptom severity scores in the PRO data have NaN values. Some patients do not have enough follow-up time to collect later time points. Like many other machine learning models, LSTM

requires complete data. As a proof of concept, we first consider three imputation methods: Linear interpolation, Kalman interpolation [13], and Multiple Imputation by Chained Equations (MICE) [3].

Both Linear and Kalman interpolations are uni-variable imputations. The first non-NA value was replicated to the start of the time series and the last-NA value was replicated to the end of the time series. Kalman smoothing requires at least three observations. When the time series contained less than 3 observations, the spline method was used to interpolate. Since the MDASI-HN ratings range from 0 to 10, imputed values that resulted in negative values were replaced by 0, and values larger than 10 were replaced by 10. Before applying the Kalman smoothing, the time series for the patient was scaled between 0 and 1, imputation applied, and then the data was scaled back to the original 0-10 scale.

MICE [3] is a multivariate imputation so the time series for all the patients are used simultaneously. That is, through an iterative series of predictive models, each specified variable in the data set is imputed using the other variables in the data set. The predictive model used to impute values can be various. In this work, we used predictive mean matching (pmm) where for each missing entry, the method forms a small set of candidates from all complete cases that have predicted values that is close to the predicted value for the missing entry and select a random candidate from the set to replace the missing value. Besides, we did 5 iterations for the imputation.

We also apply the LSTM model recursively to predict intermediate time points and then use the predicted data to train the next time step in the model.

All the data imputations described were done using only the PRO time-series data without considering any other clinical variables such as gender, age, cancer staging, or treatment.

Each imputation method produces a different complete version of the dataset. The complete data was then transformed into the correct input size of the LSTM model, and an LSTM model was trained on each of the imputed datasets. The predictions for all 28 symptoms were then compared using the Root Mean Squared Error (RMSE) and Normalized RMSE (NRMSE) metrics as defined below.

$$RMSE(\hat{\theta}) = \sqrt{MSE(\hat{\theta})} = \sqrt{E((\hat{\theta} - \theta)^2)} \quad (1)$$

where $\hat{\theta}$ is the vector of observed values of the variable being predicted and θ being the predicted values and E is the expected value.

$$NRMSE = \frac{RMSE}{y_{max} - y_{min}} \quad (2)$$

where y_{max} and y_{min} are the maximum and minimum of actual data.

4 Experimental Results

4.1 Experimental Setup

MDASI questionnaires were collected from 823 patients weekly for 6-weeks during treatment and for 3-time points after treatment (6-weeks, 6 Months, and 12 Months). The original data was split into two series: from baseline to 6-weeks after treatment, and from baseline to 12-month after treatment. We then applied the three interpolation methods on the two versions of the data and generated 6 imputed data sets. The time point to be predicted was not imputed and patients with missing surveys for that time point were excluded

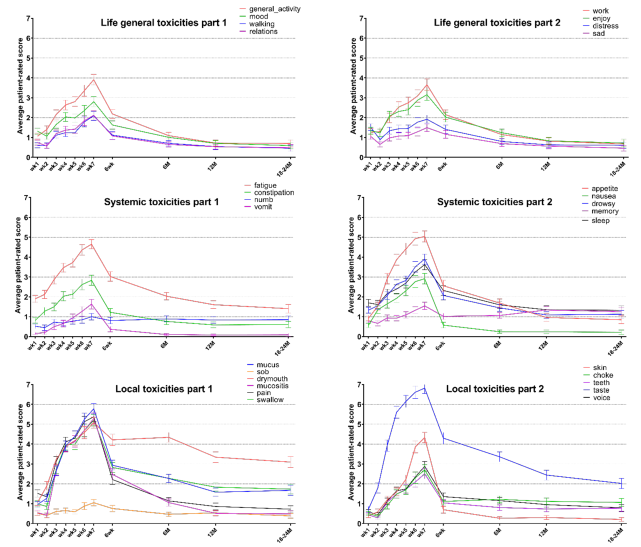


Figure 1: Average PRO scores trajectory and 95% confidence weekly during, 6 weeks, 3-6, 12 and 18-24 months after treatment for all 28 toxicities included in this study among all patients. Abbreviations: sob= shortness of breath; numb= numbness; wk=week; M=month

from the model. For testing, we only considered patients with complete data, that is, patients that completed all the PRO surveys for each time point. We ended up with a total of 651 and 483 patients predicting 6-weeks after treatment and for 12-months after treatment, respectively. All data imputation was done in R using the imputeTS [20] and mice packages. For MICE, the method used was predictive mean matching (pmm) and only one imputation was used with 5 iterations.

The LSTM models are built based on the open-source PyTorch framework. The input and output dimensions were set to 28. We used Mean Square Error (MSE) as the loss function and Stochastic Gradient Descent (SGD) as the optimizer with the learning rate of 0.215. Using a grid search for parameter tuning, we set the number of hidden layers to 1 and the number of hidden dimensions to 8. During training, we used early stopping criteria to prevent over-fitting. The RMSE score is calculated by applying the square root function on the PyTorch MSE metric and the NRMSE score is calculated by dividing the RMSE with the range (max - min) of the actual data. All the networks were run on NVIDIA GeForce RTX 2070 GPU with 8GB of memory.

4.2 PRO Data Summary

The PRO data is summarized in Figure 1 using the average symptom severity for the different time points. As can be seen, patients experience severe symptoms during the treatment and over time most of the symptoms return to baseline. However, for some symptoms and for some patients the toxicity persists even after 12 months after treatment.

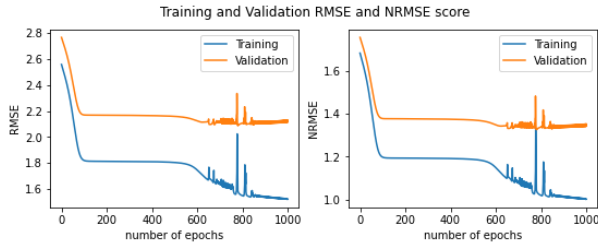


Figure 2: LSTM training and validation performance for predicting symptom severity 6-weeks after treatment using only complete data (original).

4.3 Training LSTM using only complete data

Figure 2 shows the training and validation performance for the LSTM model trained for predicting symptom severity 6-weeks after treatment when only complete data (without imputation) is used in the analysis. As can be seen, the validation RMSE score is higher than the training RMSE score all the time, which can be caused by insufficient samples in the training set. The same trend was observed for the NRMSE metric as well.

4.4 Data Imputation Evaluation

To evaluate the imputation methods, we compare the model performance in terms of RMSE and NRMSE when the LSTM was trained with complete data generated using the different imputation methods. Figure 3 shows the RMSE performance for training and validation of predicting symptom severity 6-weeks after treatment using complete datasets imputed with Linear interpolation, Kalman interpolation, MICE, and LSTM-recursive imputation methods. As can be seen, the model performance is better than when only complete data is used. Furthermore, the performance between the three imputation methods is similar and the models do not start to overfit until after 600 epochs for Linear interpolation and almost 1500 for the recursive LSTM. The final model uses the early stopping strategy to improve model performance.

Figure 4 shows the validation RMSE metric for the final models trained over the complete data (original) and the four different imputed datasets. Linear imputation, in 6Wk after prediction, has the lowest validation RMSE score of and 1.9371 among the non-recurrent imputation methods whereas the original completed data has 2.1653 RMSE. The LSTM-recursive imputation shows the best overall performance for all metrics on late symptom predictions for 6Wk and 12Mo. The LSTM-recursive has an overall lowest RMSE of 1.9142 and 1.4231 for 6Wk and 12Mo, respectively. The LSTM-recursive method also achieved the best overall performance under the NRMSE metric (not shown). Worth noting is the fact that, regardless of what imputation method was used, the models trained with imputed data, and therefore more data, performed better than the models trained with only complete data.

4.5 LSTM performance on individual symptoms

Next, we want to evaluate the LSTM performance on individual symptoms. For these experiments, we used the LSTM-recursive

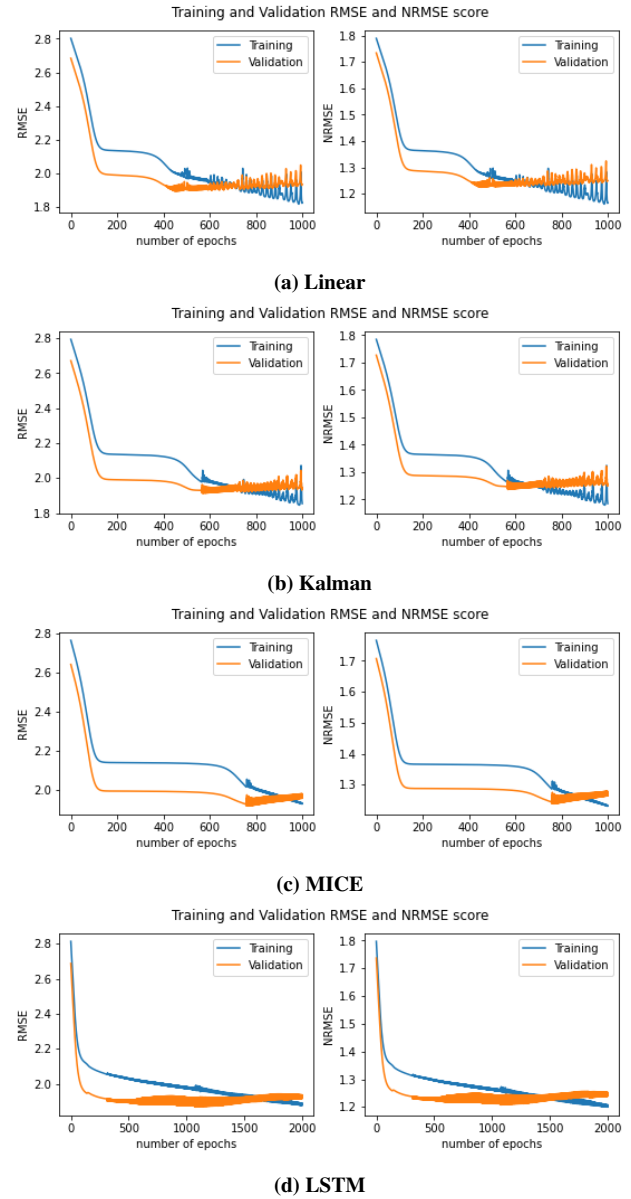


Figure 3: LSTM RMSE performance for predicting symptom severity 6 weeks after treatment using Linear, Kalman, MICE, and LSTM imputed datasets.

interpolation dataset, as it had the best overall RMSE score for all 28 symptoms combined.

Figure 5, shows the (a) RMSE and (b) NRMSE score for each symptom predicted 6-weeks and 12-month after treatment. As can be seen, symptoms like taste and dry-mouth have higher RMSE scores for predictions at both time points while other symptoms like vomit and nausea have lower RMSE scores. However, when we look at the NRMSE metrics for the same symptoms, we see that the NRMSE scores are higher for some of those symptoms with lower RMSE. The reason is that NRMSE takes into account the range of

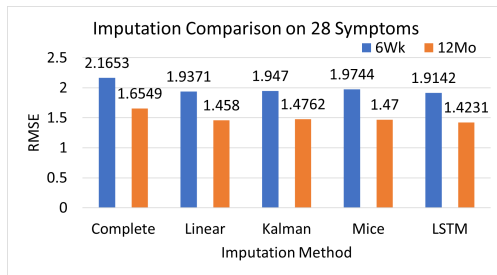


Figure 4: LSTM performance in terms of RMSE metrics for the validation set when the model is trained with only complete data (original) vs. complete data imputed with linear interpolation, Kalman interpolation, MICE, or LSTM methods.

the responses for the given time point. At 6-weeks after treatment, many symptoms still have a larger severity range than 12-months after treatment. Most of the symptoms have a lower RMSE score for 6-weeks after when compared to 12-months after treatment. The only exceptions are constipation and teeth, for which the RMSE score shows a slight increase for 12-months when compared to the 6-weeks prediction.

4.6 Comparison with other ML Models

For comparison to other machine learning models, we focus on three prevalent symptoms: pain, taste and general activity. The reason that we limit the number of symptoms is that the other models considered cannot handle multiple predictions simultaneously. We compare the LSTM performance with the performance of six other popular models: supported vector machine (SVM), K-nearest neighbour (KNN), random forest (RF), Gaussian naive Bayes (GauNB), multi-layer perceptron (MLP), and ARIMA.

Figure 6 shows the RMSE comparison for the 6-weeks after treatment prediction of pain, taste, and general activity symptoms for the 6 ML models and two LSTM: the one trained with the linear interpolated data (LSTM_L) and the recursive LSTM (LSTM*2). As can be seen, the LSTM model yields the lowest RMSE scores for all three symptoms. For pain, LSTM has an RMSE of 1.7794 which is over 20% lower than the MLP prediction, which has the second-lowest RMSE score of 2.2646. LSTM also outperformed all other models for the 12-month prediction of these symptoms but results are omitted for brevity. Interestingly, for taste and general activity, the LSTM trained over the data imputed using linear interpolation shows a slightly lower error than the recursive LSTM imputed data. A plausible explanation is that interpolation uses the before and after values in the series to impute the missing values and intuitively symptom severity follow a linear increase/decay. In contrast, LSTM only uses past information to forecast symptoms' severity. In the future, it could be worth exploring alternatives that could combine both methods.

5 Conclusion

In this work, we used the PRO data from the MDASI-HN module and applied the LSTM model to predict late toxicity from head and neck cancer treatment. An accurate prediction can help identify personalized symptom risk profiles and proactively prescribe exercises or medication that can help patients cope with symptoms to avoid

having to adjust treatment and improve the long-term quality of life of the patients.

To deal with the missing data, we applied three interpolation methods, linear, Kalman, and MICE. In addition, we also applied LSTM recursively to complete the data. We compared the performance of the LSTM model in terms of RMSE and NRMSE. The results show that using linear interpolation as the imputation method, though it is the simplest of the methods used, yielded better performance than Kalman and MICE imputations. While the LSTM imputation produces lower overall error measures than using linear interpolation, linear interpolation performed better than LSTM imputation for some individual symptoms. In all cases, the use of imputed data produced a better model than using only complete data. Furthermore, the LSTM model outperforms other machine learning models in the prediction of individual symptoms including pain, taste, and general activity symptoms.

As future work, we would like to evaluate whether the inclusion of clinical data [8, 17, 26, 29] into the analysis would further improve the predictive power of the LSTM models. There are different ways in which these clinical variables, which are mostly categorical, can be leveraged into the LSTM model to further improve model performance and symptom prediction.

6 Acknowledgments

Authors have been partially supported by NIH grants R01CA258827, R01CA225190, and R01CA214825.

References

- [1] R. Adhikari and R. K. Agrawal. An Introductory Study on Time Series Modeling and Forecasting. *CoRR*, abs/1302.6613, 2013.
- [2] A. Aktas, D. Walsh, and L. Rybicki. Symptom clusters: myth or reality? *Palliative medicine*, 24(4):373–385, 2010.
- [3] S. Buuren and C. Groothuis-Oudshoorn. MICE: Multivariate Imputation by Chained Equations in R. *Journal of Statistical Software*, 45, 12 2011.
- [4] K. Christopherson, A. Ghosh, A. Mohamed, et al. Chronic radiation-associated dysphagia in oropharyngeal cancer survivors: Towards age-adjusted dose constraints for deglutitive muscles. *Clin. Transl. Rad. Onc.*, 18:16–22, Sept. 2019.
- [5] C. S. Cleeland, T. R. Mendoza, X. S. Wang, et al. Assessing symptom distress in cancer patients: the M.D. Anderson Symptom Inventory. *Cancer*, 89(7):1634–1646, 2000.
- [6] S. J. Coons, S. Eremenco, J. J. Lundy, et al. Capturing Patient-Reported Outcome (PRO) Data Electronically: The Past, Present, and Promise of ePRO Measurement in Clinical Trials. *The patient*, 8(4), 2015.
- [7] S. T. Dong, D. S. Costa, P. N. Butow, et al. Symptom clusters in advanced cancer patients: An empirical comparison of statistical methods and the impact on quality of life. *Journal of pain and symptom management*, 51(1):88–98, 2016.
- [8] H. Elhalawani, A. S. Mohamed, A. L. White, et al. Matched computed tomography segmentation and demographic data for oropharyngeal cancer radiomics challenges. *Scientific data*, 4:170077, 2017.
- [9] S. Elsworth and S. Güttel. Time Series Forecasting Using LSTM Networks: A Symbolic Approach, 2020.
- [10] S. A. Eraj, M. K. Jomaa, C. D. Rock, et al. Long-term patient reported outcomes following radiation therapy for oropharyngeal cancer: cross-sectional assessment of a prospective symptom survey in patients ≥ 65 years old. *Rad. onc.*, 12(1), 2017.
- [11] G. Fan, L. Filipczak, and E. Chow. Symptom clusters in cancer patients: a review of the literature. *Current oncology (Toronto, Ont.)*, 14(5):173–179, 2007.
- [12] S. Hochreiter and J. Schmidhuber. Long Short-term Memory. *Neural computation*, 9:1735–80, 12 1997.
- [13] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME—Journal of Basic Engineering*, 82(Series D):35–45, 1960.
- [14] M. Kamal, M. P. Barrow, J. S. Lewin, et al. Modeling symptom drivers of oral intake in long-term head and neck cancer survivors. *Supportive care in cancer*, 27(4):1405–1415, 2019.
- [15] S. Kaushik, A. Choudhury, P. K. Sheron, et al. AI in Healthcare: Time-Series Forecasting Using Statistical, Neural, and Ensemble Architectures. *Frontiers in Big Data*, 3:4, 2020.
- [16] Z. C. Lipton, D. C. Kale, C. Elkan, et al. Learning to Diagnose with LSTM Recurrent Neural Networks, 2017.

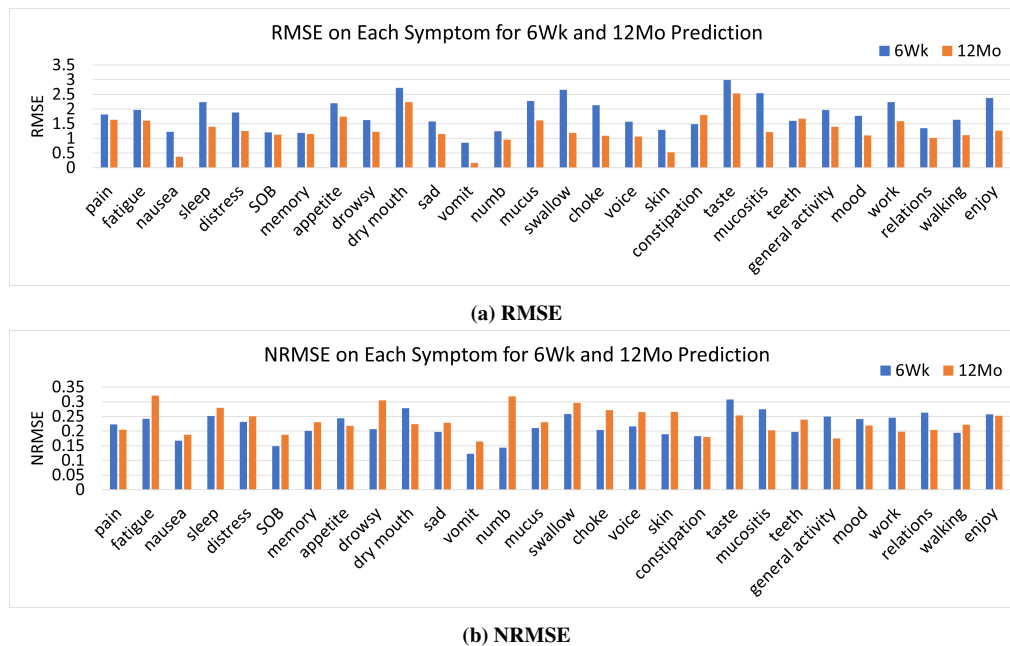


Figure 5: Validation RMSE (a) and NRMSE (b) scores for the prediction of individual symptoms 6Wk and 12Mo after treatment.

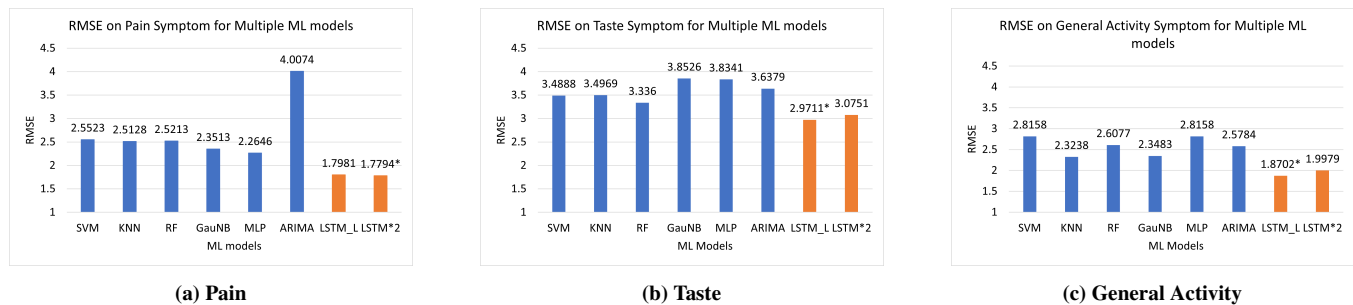


Figure 6: 6-week prediction RMSE on pain, taste, and general activity symptoms comparison for several machine learning models. Abbreviations: SVM = supported vector machine, KNN = K-nearest neighbour, RF = random forest, GauNB = Gaussian naive-bayes, MLP = multi-layer perceptron, ARIMA = Auto Regressive Integrated Moving Average, LSTM_L = LSTM on Linear Imputed data, LSTM*2 = LSTM on LSTM-recursive imputed data. * indicates the lowest RMSE score.

- [17] T. Luciani, A. Wentzel, B. Elgohari, et al. A spatial neighborhood methodology for computing and analyzing lymph node carcinoma similarity in precision medicine. *J. Biomed. Informatics*, 5:100067, 2020.
- [18] G. Maragatham and S. Devi. LSTM Model for Prediction of Heart Failure in Big Data. *J Med Syst*, 111, 2019.
- [19] G. E. Marai, C. Ma, A. T. Burks, et al. Precision Risk Analysis of Cancer Therapy with Interactive Nomograms and Survival Plots. *IEEE Trans. Vis. Comp. Graphics*, 25(4):1732–1745, 2019.
- [20] S. Moritz and T. Bartz-Beielstein. ImputeTS: Time Series Missing Value Imputation in R. *The R Journal*, 9(1):207–218, 2017.
- [21] S. Moritz, A. Sardá, T. Bartz-Beielstein, et al. Comparison of different Methods for Univariate Time Series Imputation in R, 2015.
- [22] D. I. Rosenthal, T. R. Mendoza, M. S. Chambers, et al. Measuring head and neck cancer symptom burden: the development and validation of the M. D. Anderson symptom inventory, head and neck module. *Head & neck*, 29(10):923–931, 2007.
- [23] D. I. Rosenthal, T. R. Mendoza, C. D. Fuller, et al. Patterns of symptom burden during radiotherapy or concurrent chemoradiotherapy for head and neck. *Cancer*, 120(13):1975–1984, 2014.
- [24] T. Sheu, D. Vock, A. Mohamed, et al. Conditional Survival Analysis of Patients With Locally Advanced Laryngeal Cancer: Construction of a Dynamic Risk Model and Clinical Nomogram. *Scientific Reports*, 2017.
- [25] H. M. Skerman, P. M. Yates, and D. Battistutta. Multivariate methods to identify cancer-related symptom clusters. *Res. Nursing & Health*, 32(3):345–360, 2009.
- [26] J. Tosado, L. Zdiar, H. Elhalawani, et al. Clustering of Largely Right-Censored Oropharyngeal Head and Neck Cancer Patients for Discriminative Groupings to Improve Outcome Prediction. *Scientific reports*, 10(1), 2020.
- [27] A. Wentzel, P. Hanula, T. Luciani, et al. Cohort-based T-SSIM Visual Computing for Radiation Therapy Prediction and Exploration. *IEEE Trans. Vis. and Comp. Graphics*, 26(1):949–959, Jan. 2020.
- [28] A. Wentzel, P. Hanula, L. V. van Dijk, et al. Precision toxicity correlates of tumor spatial proximity to organs at risk in cancer patients receiving intensity-modulated radiotherapy. *Radiotherapy and Oncology*, 148:245–251, 2020.
- [29] L. Zdiar, D. M. Vock, G. E. Marai, et al. Evaluating the effect of right-censored end point transformation for radiomic feature selection of data from patients with oropharyngeal cancer. *JCO clinical cancer informatics*, 2:1–19, 2018.
- [30] Z. Zhang, Y. Xie, F. Xing, et al. MDNet: A Semantically and Visually Interpretable Medical Image Diagnosis Network. *CoRR*, abs/1707.02485, 2017.

Data Management in the Data Lake: A Systematic Mapping

Firas Zouari

Lyon University, iaelyon - Lyon 3 University, LIRIS
Lyon, France
firas.zouari@univ-lyon3.fr

Khouloud Boukadi

University of Sfax
Sfax, Tunisia
khouloud.boukadi@fsegs.usf.tn

Nadia Kabachi

Univ Lyon, Université Claude Bernard Lyon 1, ERIC
Lyon, France
nadia.kabachi@univ-lyon1.fr

Chirine Ghedira-Guegan

Lyon University, iaelyon - Lyon 3 University, LIRIS
Lyon, France
chirine.ghedira-guegan@univ-lyon3.fr

ABSTRACT

The computer science community is paying more and more attention to data due to its crucial role in performing analysis and prediction. Researchers have proposed many data containers such as files, databases, data warehouses, cloud systems, and recently data lakes in the last decade. The latter enables holding data in its native format, making it suitable for performing massive data prediction, particularly for real-time application development. Although data lake is well adopted in the computer science industry, its acceptance by the research community is still in its infancy stage. This paper sheds light on existing works for performing analysis and predictions on data placed in data lakes. Our study reveals the necessary data management steps, which need to be followed in a decision process, and the requirements to be respected, namely curation, quality evaluation, privacy-preservation, and prediction. This study aims to categorize and analyze proposals related to each step mentioned above.

CCS CONCEPTS

• Information systems → Data management systems; Data warehouses; • General and reference → Surveys and overviews.

KEYWORDS

Data management, Data lake, Systematic mapping

ACM Reference Format:

Firas Zouari, Nadia Kabachi, Khouloud Boukadi, and Chirine Ghedira-Guegan. 2021. Data Management in the Data Lake: A Systematic Mapping. In *25th International Database Engineering Applications Symposium (IDEAS 2021)*, July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 5 pages. <https://doi.org/10.1145/3472163.3472173>

1 INTRODUCTION

The last few decades have seen the emergence of several technologies related to data and their management. Consequently, many

structures were proposed to hold, regroup and query the data like files, databases, data warehouses, and recently data lakes. According to Fang [7], the data lake is a methodology created by internet companies to handle the scale of web data and to perform new types. It can be described as a massive data repository based on low-cost technologies that improves the capture, refinement, archival, and exploration of raw data within enterprise transformations. This emergent technology is characterized by its ability to carry massive data in a very heterogeneous form (i.e., non-structured, semi-structured, and structured) at the same time [13]. This valuable characteristic had increased more and more the data lake usage, especially for putting in place real-time applications when there is a time constraint preventing from performing a process of unifying data schemes before loading data to a repository.

Unfortunately, despite the bright side of the data lake, its adoption faces many difficulties for several reasons. First, ingesting data from multi-sources raises many questions about the quality of the ingested data. Data quality may be described as "fitness for use" which consists of assessing the quality of the data according to its context of use. Data quality may be appropriate for one use, but may not be of sufficient quality for another use [16]. In addition to data and source quality, ensuring the privacy of data carried in the data lake is raised by professionals working with sensitive data (e.g., healthcare, governmental, social, etc.). Privacy can be defined as "the claim of individuals, groups, or institutions to decide for themselves when, how and to what extent information about them is communicated to others" [11]. To overcome these challenges, scientists have started to find mechanisms to curate data, ensure their quality and preserve privacy. This work aims to categorize the existing works and to identify the open issues regarding data management in a data lake. Thus, we consider that it is necessary to apply first comprehensive analysis of the mechanisms used to manage data in a data lake and perform research using curated data while ensuring the quality of data and preserving their providers' privacy.

For this purpose, we propose in this paper a systematic mapping to provide a classification scheme of articles by applying a systematic mapping method [15] which consists of the following five inter-dependent steps: (i) Definition of the research scope, (ii) Conduct search to identify all papers, (iii) Screening of documents to select the relevant ones, (iv) Definition of a classification scheme and (v) Sorting papers according to the classification scheme. After

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Association for Computing Machinery.

ACM ISBN 978-1-4503-8991-4/21/07.

<https://doi.org/10.1145/3472163.3472173>

that, we discuss the obtained results and shed light on open issues related to data management in data lakes.

This paper is organized as follows: Section 2 depicts a scenario of performing predictions using data stored in a data lake. Section 3 presents the systematic mapping method and finally, Section 4 proposes a discussion of the systematic mapping results and the open issues.

2 APPLICATION SCENARIO

To motivate the use of data lake for data prediction, we consider the scenario of disease prediction and recommendation of actions to be taken to manage a crisis situation. The disease control and prevention center wants to take preventive actions after identifying a new unknown virus. The detected virus spread quickly and could be considered as a high risk for many countries. However, disease prediction and required actions recommendation need to be taken in real-time or near real-time on multi-source of data, that can be sometimes complex. Therefore, there is a need to store data into a data container that can hold heterogeneous data sources and perform predictions in a very brief delay. The disease prevention and control center can use a crisis analysis and management system to perform data analysis and generate predictions and health recommendations. The data ingested into this system can be carried into a data lake. Once a user's query is received, the system is supposed to perform a data curation and integration process to predict the disease and then generate suitable recommendations. Such a system should maintain a degree of intelligence and adaptability to fit the dynamic change of the global context (i.e., emergency, crisis, etc.). Furthermore, it should adapt to the user's context, namely the purpose expected by the user. Such a system has to be also a multi-purpose system that may be used by multiple users for different purposes.

We assume that the system can be made up of the following components: data ingestion, curation, integration, and prediction. These components represent the data management steps in a data lake. In what follows, we present the challenges related to each data management step using a systematic mapping process.

3 SYSTEMATIC MAPPING

To have a global view and analyze curation, quality evaluation, privacy-preserving, and prediction using data placed in the data lake, we performed a systematic mapping as defined in [15] and described hereafter.

3.1 Step 1: Definition of Research Scope

The first step consists of defining research questions to query. As aforementioned, the aim of our study is (i) to categorize and quantify research contributions to data curation in the data lake (ii) to categorize research contributions that considered data and source quality assessment and privacy-preserving in the data lake (iii) to categorize the studies that made predictions based on data carried out in the data lake, and (iv) to discover open issues and limitations in existing works. Thus, our study is guided by the questions mentioned in Table 1.

Table 1: Research questions for the systematic mapping

Research Question	Aim
RQ1: What are the existing proposals for data curation in a data lake?	This question helps to identify the existing approaches for enrichment, cleaning, linking, and structuring of data to get a well-organized data lake.
RQ2: What are the proposals for data and source quality evaluation and privacy-preserving?	This question aims at discovering the existing approaches of data quality evaluation and privacy-preserving to guarantee the quality of predictions and the anonymity of data providers.
RQ3: How did the published papers deal with prediction and reasoning?	This question helps to identify the proposed prediction approaches using data sources placed in the data lake.

3.2 Step 2: Search Conducting

Search conducting consists of collecting papers from relevant scientific search engines/ databases, namely ACM Digital Library¹, IEEE Xplore², ScienceDirect³, and Springer⁴. We have chosen a set of keywords to retrieve papers from databases. We used the following query, which is divided into three sub-queries:

("Data Lake") AND (Curation OR Enrichment OR Cleaning)
 ("Data Lake") AND (Quality OR Privacy AND Preserving)
 ("Data Lake") AND (Prediction OR Reasoning)

At the end of this step, we found 1880 publications.

3.3 Step 3: Paper Screening

Following search conducting, we apply paper screening by defining a set of inclusion and exclusion criteria are defined. We considered only papers in English and treating data lakes by proposing an original work such as an approach, a method, a framework, etc. We excluded tables of contents, summaries, abstracts, forewords, encyclopedias, books, editorials, reports, studies, reviews, duplicates, and informative papers. By applying the filtering process, only 98 papers⁵ were included while 1782 were excluded.

3.4 Step 4: Keywording Using Abstracts

This step consists of defining the classification scheme. First, we define the facets that combine the classification dimensions. Afterward, we consider the relevant frequent words in papers' abstracts as dimensions. We define six facets for classifying challenges in data lake:

- **Application domains:** it deals with the field in which the authors present their proposals, such as agriculture, biology,

¹<https://dl.acm.org/>

²<https://ieeexplore.ieee.org/Xplore/home.jsp>

³<https://www.sciencedirect.com/>

⁴<https://link.springer.com/>

⁵https://zenodo.org/record/4749337#.YJqT_bUzZPY

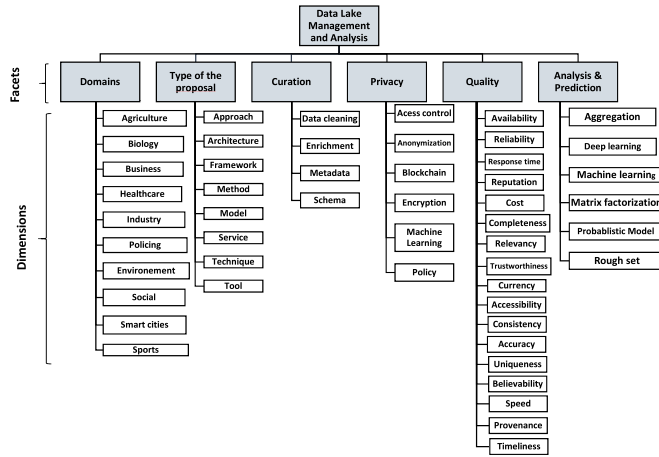


Figure 1: Classification scheme facets and dimensions

business, healthcare, industry, policing, environment, social, smart cities, and sports.

- **Type of the proposal:** it concerns the proposal's type, such as approach, architecture, framework, method, model, service, technique, and tool.
- **Curation:** it reveals the proposed curation technique, such as data cleaning, enrichment, metadata, and schema.
- **Privacy:** it concerns the used technique for privacy-preserving, such as access control, anonymization, blockchain, encryption, machine learning, and policy.
- **Quality:** it deals with the quality dimensions considered in the proposal, such as availability, reliability, response time, reputation, cost, completeness, relevancy, trustworthiness, currency, accessibility, consistency, accuracy, uniqueness, believability, speed, provenance, and timeliness.
- **Analysis & prediction:** it mentions the used technique for data analysis and prediction, such as aggregation, deep learning, machine learning, matrix factorization, probabilistic model, and rough set.

3.5 Step 5: Data Extraction and Mapping Process

In this step, we perform the mapping process to provide answers to our research questions. Thus, we combine the facets, and we present the results in bubble charts⁶.

RQ1: What are the existing proposals for data curation in a data lake?

By combining contribution with curation facets, we observe that most of the papers are presenting approaches (67.80 %). We noticed that more than one-third of the contributions are addressing the enrichment. The identified proposals are also handling Schema (30.51%) and Metadata (16.95%). While the least addressed dimension is Data Cleaning (10.17%). Thus, data enrichment and schema curation are the most addressed aspects.

⁶<https://zenodo.org/record/5014065#.YNIXFegZPZ>

RQ2: What are the proposals for data and source quality evaluation and privacy-preserving?

We have combined the contribution facet with the quality and the privacy facets. We found that the most addressed quality dimension is completeness (21.21%) followed by provenance (12.12%). We think this can be explained by the nature of data stored in data lakes that are often incomplete and can lead to uncertainty. At the same time, the papers paid less attention to the other quality dimensions. As for privacy mechanisms, the most used are Blockchain (25%), Encryption (20%), and Access Control (20%), while the least addressed are Policy and Machine Learning (10%). We point out that quality dimensions like completeness and provenance are attracting researchers' attention. As for privacy, blockchain is a promising research field.

RQ3: How have the published papers treated prediction and reasoning?

To answer this question, we have combined the prediction and the domain facets. We noticed that machine learning is the most used mechanism to analyze and predict, while there is no significant difference between the other mechanisms. Most of the contributions did not address a specific field, but the healthcare (24.24%) field had more attention than the other areas like sports (3.03%) and social domain (3.03%). We find that machine learning is the most used technique for prediction. Regarding application fields, several contributions are addressing the healthcare field, while most of the proposals did not imply an application field.

4 DISCUSSION AND OPEN ISSUES

Our systematic mapping shows that the proposals address the data placed in data lakes from different data management perspectives, such as curation, quality evaluation, privacy-preserving, and prediction. In this section, we present the open issues of each dimension.

4.1 Curation

After establishing the statistical study, we noticed that most of the contributions address data curation issues in a data lake compared to other dimensions such as quality assessment, privacy, and prediction. We previously outlined that data and source curation is a fundamental step in data lake's data management (see Section 2). Since data sources are stored in the data lake in their native format, they need to be pretreated to be exploited in the following steps (i.e., integration and prediction). We classify curation tasks into three classes: (1) Data Curation, including data enrichment such as semantic enrichment and contextualization. Also, it includes data cleaning, which encompasses data deduplication and data repair. (2) Curation tasks related to Metadata such as metadata extraction and metadata modeling (3) Curation tasks related to schema like extraction, matching, mapping, and evolution. We noticed that we cannot apply all curation approaches to all data types. Thus, some curation approaches like schema extraction are restricted to semi-structured and unstructured data. Regarding curation, we noticed that the use of machine learning techniques is still limited, especially for data cleaning and schema mapping. Thus, several curation tasks such as deduplication, spotting errors, and violation and repairing data are hard to be automated. Indeed, most of the studied curation approaches rely on rule-based techniques, semantic techniques, or the

incorporation of machine learning with one of the above techniques. Consequently, we identified rule-based contributions ensuring curation tasks like detecting violations such as [9]. Otherwise, we found that machine learning techniques are combined with other methods such as crowdsourcing to ensure the curation task like the work presented in [3]. The latter proposes a semi-automatic approach composed of automatic curation via curation services and manual annotations via crowdsourcing and experts' annotations. The proposed curation services [4] representing a curation step (e.g., Linking dataset with knowledge base). These services rely on different techniques like rules, dictionaries, ontologies, and machine learning. Thus, authors have ensured the automatic completion of some curation tasks (i.e., via orchestrating curation services), but they still need human intervention. The incorporation of machine learning with other techniques can be explained by the subjective aspect of some curation tasks like detecting errors, which cannot be identified using a series of rules and require human intervention. Thus, full autonomy in performing curation and the use of machine learning and rules generalization are still open issues of curation.

4.2 Quality

Our systematic mapping has also considered the quality aspect. Data/source quality evaluation can be performed in the integration step, mainly when discovering data, which consists of finding a subset S of relevant data sources among the organization's many sources, appropriate to a user-supplied discovery query [5]. Our study shows that the authors focused mainly on completeness and provenance dimensions due to data incompleteness and uncertainty. Thus, they try to assess or improve the quality of these dimensions, primarily. The data quality depends on conducted curation tasks. We state that the authors in [2] insisted on data completeness due to its importance in data analytics. Thus, they propose a data imputation approach to repair erroneous data that enhances analysis outcomes quality. Accordingly, while performing erroneous data repair, this negatively impacts the accuracy dimension. Similarly, achieving missing data repair negatively influence completeness and accuracy dimensions. The quality evaluation is not enough by itself. Indeed, to perform the quality evaluation, a conceptual background is needed to identify which information must be of high quality and the quality dimensions to be used for assessment [14]. Moreover, we identified variability and diversity of subjective and objective quality dimensions required to evaluate data and its source [12, 14]. Usually, the quality dimensions are defined at design time and imply a domain expert's involvement to identify the required quality dimensions for a particular data and data source, which is time-consuming and error-prone. Therefore, we think that implementing autonomous systems helps to identify the necessary quality dimensions at run-time. Such systems can adapt to several factors, such as data source characteristics (e.g., data source format) and user needs. We assume that the system described in Section 2 is used by several users' roles. For example, we distinguish two users ensuring different missions in a pandemic situation. Thus, their requirements and the expected timely data quality differ. Accordingly, we think that an adaptive system can optimize the data evaluation process in terms of execution time and alignment with user expectations.

4.3 Privacy

Privacy is of great importance not only to internet users but also to businesses, legal communities, and policymakers. Hence, managers can seek to predict which privacy-enhancing initiative so that to gain a competitive advantage [1]. Therefore, it is important to include privacy-preserving mechanisms in each decision process that contains steps of collecting, accessing, and storing individuals' data. As for data lakes, they are designed to carry various types of data sources coming from different data sources. Data can be internal or external to an organization and can also be sensitive. Data in different fields like healthcare and biology are still represented as data silos due to their sensitivity. Users are hence still afraid of leaking their data, such as medical analyzes and genetic information. Thus, a privacy-preserving mechanism is needed to prevent data leakage threats, especially for systems that carry individuals' information like the one presented in Section 2. In the presented scenario, individuals data are collected from various sources such as hospitals, health institutes, government, smartphones, wearable health devices, and IoT. Thus, the collected data are mostly sensitive and must be protected from the above threats. Our study reveals that various techniques can be used to preserve privacy in data lakes and big data ecosystems, such as access control, anonymization, blockchain, encryption, machine learning, and the definition of policies. We think that, for systems similar to the presented one (Section 2), it is preliminary to involve access control mechanisms to grant permissions according to users' roles. However, it also requires more sophisticated mechanisms to ensure anonymization and data encryption. However, when considering massive data, anonymization techniques may face a linking problem with original datasets. Regarding encryption, this technique may need important computational performances when dealing with a massive amount of data [8]. Thus, we think that these techniques should be readjusted to fit massive data processing. According to our study, the usage percentage of privacy-preservation techniques is almost close, where the most used one is blockchain. Since this latter is an emergent technology, it had received more interest from the scientific community. Thus, blockchain data is perfect for data analysis as it is secure and valuable. Blockchain would be an alternative to conventional privacy systems, such as access controls, which provide more convenience and security for users to disclose their personal information. However, according to our study, the use of this mechanism is still limited in data lakes. We think that it can be subject to many contributions in the future.

4.4 Prediction

The conducted study shows that data lakes offer a sophisticated infrastructure to perform predictions. Many companies propose a data lakes architecture as a service to visualize, extract dashboards and perform machine learning, such as Microsoft Azure and Amazon AWS.

Our study reveals that the most proposed contributions for prediction are generic and not restricted to a specific field, whereas the healthcare field has taken more attention than other areas. It can be explained by the nature of healthcare data, which can vary in its format (i.e., analyzes, medical images, patient records, etc.).

Data lakes can be more efficient in developing real-time applications. Contrary to data warehouses, data are stored in data lakes using the Extract-Load-Transform (ELT) process instead of the Extract-Transform-Load (ETL) process. Thus, data lakes extract schema-on-read instead of schema-on-write. Postponement of the transformation step can save time, making data lakes more suitable to perform real-time predictions. Therefore, we think that it is useful to use data lakes to achieve predictions and reach such optimization degrees. Since the data lake contains a huge amount of data, we also believe that it is necessary to improve the techniques used for data integration and prediction to cope with this volumetry. Furthermore, we think that the variety of data formats and structures may influence the prediction process. As presented in Section 2, data is ingested to data lake from different sources like health institutes, wearable health devices, social media, and smartphones. Thus, the data lake holds very heterogeneous data sources such as relational databases, images, text files, captured signals, audio files, etc. These datasets are used during the prediction process to generate useful outcomes for the users. To the best of our knowledge, several predictors have proven their performances with a specific data format. For example, CNN is a deep learning architecture that is popular for image and video classification [6]. However, it may not be the most convenient predictor to use with another data format. Besides, datasets may concern different topics. Thus, the latter may influence the generated outcomes and their interpretability. For example, dealing with users Tweets differs from dealing with chest X-ray images. Thus, we think that prediction and interpretability generalization is a key challenge. On the other hand, we notice that ensemble learning is gaining more interest in recent years [10] that can be valuable in such a context. Ensemble methods like stacking allow the combination of multiple individual predictors for the final prediction [17]. Hence, We think that stacking is a promising method and is convenient for prediction using data stored in data lakes.

5 CONCLUSION

Since the data lake is still in its early stages, few reviews are proposed in the literature. These works have adequately defined the scope of the study. Nonetheless, they are based on a few papers and do not present a statistical analysis of the covered works nor a systematic structure. We strongly agree that the current state of the art in data lakes needs to be thoroughly examined to determine statistics on what has been completed and what remains to be carried out. Our systematic mapping, the proposal of this paper, is a step in this direction. It broadens the breadth of the research by providing a deep analysis of the data lake and its related concepts for decision system implementation. Thus, we categorized and studied works related to data management in data lakes, which assisted us in identifying open issues for curation, quality evaluation, privacy-preserving, and prediction for data hosted in a data lake. Among the findings is that the use of machine learning and generalized rules for curation remains limited. To the best of our knowledge, the current proposals lack autonomy in curating various data sources. Similarly, when it comes to quality evaluation, the current proposals cannot identify the required quality dimensions to judge the "fitness for use" of such data autonomously. In

terms of prediction, we believe that the data lake is valuable for prediction, particularly for real-time applications, due to its assets. Furthermore, we discovered few contributions addressing privacy-preserving in data lakes. However, we believe that the techniques used for privacy-preserving and prediction need to be improved to deal with the massive amounts of data in data lakes. As a result, it is critical to propose contributions in this direction. Following this study, we hope that our findings will assist researchers working on data lakes to identify existing gaps in each of the dimensions mentioned above.

REFERENCES

- [1] ACQUISTI, A., JOHN, L. K., AND LOEWENSTEIN, G. What is privacy worth? *Journal of Legal Studies* 42, 2 (2013), 249–274.
- [2] AHMADOV, A., THIELE, M., EBERIUS, J., LEHNER, W., AND WREMBEL, R. Towards a Hybrid Imputation Approach Using Web Tables. *Proceedings - 2015 2nd IEEE/ACM International Symposium on Big Data Computing, BDC 2015*, MI (2016), 21–30.
- [3] BEHESHTI, A., BENATALLAH, B., NOURI, R., AND TABEBORDBAR, A. CoreKG: a Knowledge Lake Service. *Knowledge Lake Service. PVLDB* 11, 12 (2018), 1942–1945.
- [4] BEHESHTI, S. M. R., TABEBORDBAR, A., BENATALLAH, B., AND NOURI, R. On automating basic data curation tasks. *26th International World Wide Web Conference 2017, WWW 2017 Companion* (2019), 165–169.
- [5] CASTRO FERNANDEZ, R., ABEDJAN, Z., KOKO, F., YUAN, G., MADDEN, S., AND STONEBRAKER, M. Aurum: a data discovery system. *Proceedings - IEEE 34th International Conference on Data Engineering, ICDE 2018* (2018), 1001–1012.
- [6] DIWAKAR, M., TRIPATHI, A., JOSHI, K., MEMORIA, M., SINGH, P., AND KUMAR, N. Latest trends on heart disease prediction using machine learning and image fusion. *Materials Today: Proceedings* 37, Part 2 (2020), 3213–3218.
- [7] FANG, H. Managing data lakes in big data era: What's a data lake and why has it become popular in data management ecosystem. In *2015 IEEE International Conference on Cyber Technology in Automation, Control and Intelligent Systems, IEEE-CYBER 2015* (2015), Institute of Electrical and Electronics Engineers Inc., pp. 820–824.
- [8] HASSAN, M. U., REHMANI, M. H., AND CHEN, J. Privacy preservation in blockchain based IoT systems: Integration issues, prospects, challenges, and future research directions. *Future Generation Computer Systems* 97 (2019), 512–529.
- [9] KONSTANTINOU, N., ABEL, E., BELLOMARINI, L., BOGATU, A., CIVILLI, C., IRFANIE, E., KOEHLER, M., MAZILU, L., SALLINGER, E., FERNANDES, A. A., GOTTLOB, G., KEANE, J. A., AND PATON, N. W. VADA: an architecture for end user informed data preparation. *Journal of Big Data* 6, 1 (2019), 1–32.
- [10] KRISTIANI, E., CHEN, Y. A., YANG, C. T., HUANG, C. Y., TSAN, Y. T., AND CHAN, W. C. Using deep ensemble for influenza-like illness consultation rate prediction. *Future Generation Computer Systems* 117 (2021), 369–386.
- [11] LOUKIL, F., GHEDIRA-GUEGAN, C., BENHARKAT, A. N., BOUKADI, K., AND MAAMAR, Z. Privacy-aware in the IoT applications: A systematic literature review. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (2017), vol. 10573 LNCS, Springer Verlag, pp. 552–569.
- [12] NAGARAJAN, R., AND THIRUNAVUKARASU, R. A Service Context-Aware QoS Prediction and Recommendation of Cloud Infrastructure Services. *Arabian Journal for Science and Engineering* 45, 4 (2020), 2929–2943.
- [13] NARGESIAN, F., ZHU, E., MILLER, R. J., PU, K. Q., AND AROCENA, P. C. Data lake management: Challenges and opportunities. *Proceedings of the VLDB Endowment* 12, 12 (2018), 1986–1989.
- [14] PALACIO, A. L., LÓPEZ, Ó. P., AND RÓDENAS, J. C. C. A method to identify relevant genome data: Conceptual modeling for the medicine of precision. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (2018), vol. 11157 LNCS, Springer Verlag, pp. 597–609.
- [15] PETERSEN, K., FELDT, R., MUJTABA, S., AND MATTSOON, M. Systematic mapping studies in software engineering. Tech. rep., 2008.
- [16] YAYI, G. K., AND BALLOU, D. P. Examining Data Quality. *Communications of the ACM* 41, 2 (1998), 54–57.
- [17] WANG, L., ZHU, Z., SASSOUBRE, L., YU, G., LIAO, C., HU, Q., AND WANG, Y. Improving the robustness of beach water quality modeling using an ensemble machine learning approach. *Science of the Total Environment* 765 (2021), 142760.

Rigorous Measurement Model for Validity of Big Data: MEGA Approach

Dave Bhardwaj

Computer Science & Software Engineering Dept.
Concordia University
Montreal, Canada
dave.bhardwaj@mail.concordia.ca

Olga Ormandjieva

Computer Science & Software Engineering Dept.
Concordia University
Montreal, Canada
olga.ormandjieva@concordia.ca

ABSTRACT

Big Data is becoming a substantial part of the decision-making processes in both industry and academia, especially in areas where Big Data may have a profound impact on businesses and society. However, as more data is being processed, data quality is becoming a genuine issue that negatively affects credibility of the systems we build because of the lack of visibility and transparency on the underlying data. Therefore, Big Data quality measurement is becoming increasingly necessary in assessing whether data can serve its purpose in a particular context (such as Big Data analytics, for example). This research addresses Big Data quality measurement modelling and automation by proposing a novel quality measurement framework for Big Data (MEGA) that objectively assesses the underlying quality characteristics of Big Data (also known as the V's of Big Data) at each step of the Big Data Pipelines. Five of the Big Data V's (Volume, Variety, Velocity, Veracity and Validity) are currently automated by the MEGA framework. In this paper, a new theoretically valid quality measurement model is proposed for an essential quality characteristic of Big Data, called Validity. The proposed measurement information model for Validity of Big Data is a hierarchy of 4 derived measures / indicators and 5 based measures. Validity measurement is illustrated on a running example.

CCS CONCEPTS

• Software and its engineering • Software and its engineering ~ Software organization and properties

KEYWORDS

Big Data quality characteristics (V's), Validity, measurement hierarchical model, representational theory of measurement.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada
© 2021 Copyright is held by the owner/author(s). Publication rights licensed to ACM. ACM 978-1-4503-8991-4/21/07...\$15.00
<https://doi.org/10.1145/3472163.3472171>

ACM Reference format:

Dave Bhardwaj, Olga Ormandjieva. 2021. Rigorous Measurement Model for Validity of Big Data: MEGA Approach. In *Proceedings of ACM 25th International Database Engineering & Applications Symposium (IDEAS'21)*. ACM, New York, NY, USA, 7 pages. <https://doi.org/10.1145/3472163.3472171>

1 Introduction

Big data refers to the vast amount of digital data stored and originated from different sources of digital and physical environments. Big Data analysis has increased businesses' ability to gain deeper understanding of their customers' preferences and to focus their resources on rising profits [1]. A large variety of industries benefit greatly from the use of Big Data, from the healthcare industry to agriculture and farming ([2][3][4][5][6]). For instance, being able to model and predict health assessment from electronic health records is one of the many kinds of advancements that can be achieved by leveraging Big Data in the healthcare space ([6][7]).

Big Data analysis and interpretation depend highly on the quality of underlying data as an eminent factor for its maturity. Data isn't always perfect and building models of data that hasn't been properly assessed can lead to costly mistakes [8]. Hence, models that are built off processed Big Data can only be as good as the quality of the underlying data. Therefore, as Big Data Pipelines begin to handle larger amounts of data, the need for modeling and monitoring data quality becomes indispensable for the stakeholders involved. This need is addressed in this research by proposing a novel quality measurement framework for Big Data (MEGA) for modeling and monitoring the quality characteristics of Big Data (the V's.), where data issues can be identified and analyzed continuously by integrating data quality measurement procedures within Big Data Pipelines phases [9]. The goal is to flag data quality issues before they propagate into the decision-making process [9].

MEGA framework focuses on ten of the intrinsic Big Data quality characteristics referred to as 10 V's, namely: Volume, Variety, Velocity, Veracity, Vincularity, Validity, Value, Volatility, Valence and Vitality. Measurement information models were published for Volume, Velocity, Variety and Veracity ([10][11]).

In this paper we propose a new hierarchical measurement of the Big Data quality characteristic referred to as Validity. The proposed measurement model is built upon; i) the NIST (National Institute of Standards and Technology) taxonomy towards to the standardization of big data technology [12], ii) measurement

principles described in ISO/IEC/IEEE Std. 15939 [13], and iii) the hierarchical measurement models discussed in [10] and [11]. The newly proposed Validity measurements are validated theoretically using the representational theory of measurement [14]. It is to be noted that the remaining V's (Vincularity, Value, Volatility, Valence and Vitality) targeted by MEGA framework are outside the scope of this paper and will be tackled in our future work.

The rest of the paper is organized as follows: Section 2 summarizes the background knowledge needed to understand the rest of the paper. In Section 3, the proposed measurement model for Validity of Big Data is compared with the related work in the field and described in detail. The indicators of Validity's characteristics Accuracy, Credibility and Compliance are explained in sections 4, 5 and 6 (this includes describing the many base measures and derived measures needed to construct these indicators for Big Data). All measurements are illustrated on a running example and validated theoretically. The Validity indicator and its measurement hierarchy are depicted graphically in section 7. Section 8 concludes this paper and outlines our future research directions.

2 Background: MEGA Framework

Big data analysis and interpretation depend highly on the quality of underlying data as an eminent factor for its maturity. In this research we tackle the Big Data's inherent quality characteristics known as the V's of Big Data [9]. The MEGA framework automates the V's measurement information models aimed at evaluating the quality aspects of Big Data. The MEGA approach aims to solve problems that include the ability to: i) process both structured and unstructured data, ii) track a variety of base measures depending on the quality indicators defined for the V's, iii) flag datasets that pass a certain quality threshold, and iv) define a general infrastructure for collecting, analyzing and reporting the V's measurement results for trustable and meaningful decision-making.

2.1 The MEGA Architecture

The MEGA architecture focuses on having quality policies that are flexible enough to target a variety of Big Data projects. It allows data engineers and users to select V's according to their needs and provides them with more flexibility, while data is being evaluated by the Quality layer. The MEGA solution allows for data to travel from the left to the right of the Big Data Pipeline unimpeded according to the scheduling protocol. This permits the Big Data Quality layer to measure attributes in parallel, while the pipeline continues processing data. The Big Data Quality layer permits the user to halt the pipeline process until quality validation is completed, by specifying constraints in the Quality Policy Manager. The goal is to allow the user the most amount of freedom in terms of adapting the Big Data quality assessment to their specific needs at each phase of the pipeline.

The MEGA framework currently supports the 3V's measurement information model [10], the Veracity measurement information model [11], and the newly proposed Validity measurement information model described from section 3. For more details on the proposed MEGA architecture and its comparison with the related work ([16][17][18][19]) please refer to [9].

2.2 NIST Taxonomy of Big Data and ISO 15939

In this section we describe the foundation of the Validity measurement model built upon the NIST taxonomy [12], in which a hierarchy of roles/actors and activities including data elements as the smallest level, records as groups of elements, DS as groups of records and finally MDS.

Data elements (DE) are considered as the smallest level and are populated by their actual value, constrained by its data type definition (e.g.: numeric, string, date) and chosen data formats. A data element can refer to a single token such as a word in the context of unstructured text, for example.

Records (Rec) are in the form of structured, semi-structured and unstructured. Mobile and Web data (e.g.: online texts, images and videos) are examples of unstructured data.

DataSets (DS) are considered as groups of records, and **Multiple DataSets (MDS)** are groups of DS with the emphasis on the integration and fuse of data. The model defines two types of measures: base measures and derived measures [13].

A **base measure** is defined in ISO/IEC/IEEE Std. 15939 as functionally independent of other measures.

A **derived measure** is defined as a measurement function of two or more values of base/derived measures.

The novel Validity measurement information model proposed in this work defines how the relevant attributes are quantified and converted to indicators that provide a basis for decision-making. The Validity measures built hierarchically upon the model specified in section 2.2 are described in sections 3, 4, 5, 6 and 7.

2.3 Representational Approach to Validation

Validation is critical to the success of Big Data measurement. Measurement validation is "the act or process of ensuring that (a measure) reliably predicts or assesses a quality factor" [14]. The Validity measures are theoretically validated using the Representational Theory of measurement [14] with respect to Tracking and Consistency criteria introduced in [15].

Tracking Criterion assesses whether a measurement is capable of tracking changes in product or process quality over the life cycle of that product or process.

Consistency Criterion assesses whether there is a consistency between the ranks of the characteristics of big data quality (V's) and the ranks of the measurement values of the corresponding indicator for the same set. The change of ranks should be in the same direction in both quality characteristics and measurement values, that is, the order of preference of the Validity will be preserved in the measurement data.

Tracking and consistency are a way to validate the representational condition without collecting and analyzing large amounts of measurement data, thus can be done manually.

2.4 Overview of the 3V's and Veracity

The MEGA framework automates the 3V's measurement information model proposed to quantify three aspects of Big Data – Volume, Velocity and Variety. Four levels of entities have been considered, derived from the underlying Big Data interoperability

framework NIST (National Institute of Standards and Technology) standard hierarchy of roles/actors and activities [12]. The 3V's measures were validated theoretically based on the representational theory of measurement. For more details, please refer to [10]. Veracity is one of the characteristics of big data that complements the 3V's of Big Data and refers to availability, accuracy, credibility, correctness and currentness quality characteristics of data defined in ISO/IEC DIS 25024 [20]. A measurement information model for Veracity of big data was built upon [10] and published in [11].

3 Measurement Information Model for Validity

Validity of Big Data is defined in terms of its accuracy and correctness for the purpose of usage [20]. However, few studies have been done on the evaluation of data validity.

3.1 Comparison with Related Work

Big Data validity is measured in [21] from the perspective of completeness, correctness, and compatibility. It is used to indicate whether data meets the user-defined condition or falls within a user-defined range. The model proposed in [21] for measuring Validity is based on medium logic. In contrast, in our work we consider a 3-fold root cause of Validity inspired by the notions of ISO/ 25024 data quality characteristics accuracy, credibility and compliance: i) the accuracy of data in MDS, ii) the credibility of DS in MDS, and iii) the compliance of data elements in records, compliance of records in DS, and compliance of DS in MDS.

3.2 Mapping of Validity to ISO/IEC DIC 25024

Big Data validity is measured in this paper from the perspective of accuracy, credibility, and compliance, which are adapted and redefined in order to provide an evaluation of the Big Data Validity with respect to defined information needs of its measurement model (see Figure 1).

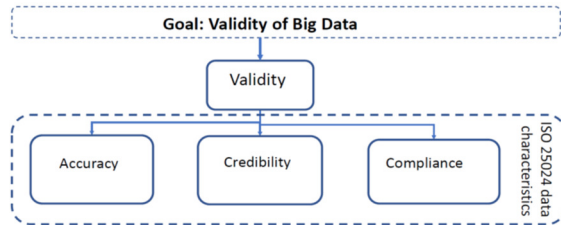


Figure 1: Big Data Validity Mapping to ISO25024

The measurement information model for the Validity of Big Data defines 3 indicators for measuring accuracy, credibility, and compliance, as described in sections 4, 5 and 6.

4 Accuracy Indicator (Acc)

Big data accuracy is essential for the Validity of Big Data. The users of big data sets require the highest validity of their data, but it's a well-known fact that big data is never 100% accurate.

4.1 Notion of Accuracy

According to the dictionary definition, accuracy means “the quality or state of being correct or precise”. ISO/IEC DIC 25024 defines data accuracy as a degree to which “data has attributes that correctly represent the true value of the intended attribute of a concept or event in a specific context of use.” It also states that accuracy can be measured from the “inherent” point of view only. One of the ways to increase the accuracy is to match records and merge them if they relate to the common values of the data attributes. Consequently, we define the accuracy as a measure of the common information in DS relationships within MDS.

4.2 Base Measures and Derived Measures

To quantify objectively the common information within the multiple datasets (MDS), we use Emden's information theory model [22]: we first abstract the MDS as an *Attribute-Record* table, where the rows represent all data elements in MDS, and the columns represent the records in MDS. The value of a *cell_{ij}* of the resulting *Attribute-Record* table is set to '1', if the data element is included in the record; otherwise, the value of the cell is '0'.

Base Measures H_{acc} and H_{max} . We use the notion of entropy H to quantify objectively the common information [22]. The measurement formula for calculating the entropy H_{acc} in the Accuracy measurement model is as follows:

$$H_{acc} = \log_2(Lbd) - 1 / Lbd * \sum_{j=[1..k]} p_j \log_2(p_j) \quad (1)$$

where Lbd is the count of the total number of records in the MDS, k is the number of different column configurations in the *Attribute-Record* table corresponding to MDS, and p_j is the number of columns with the same configuration so that,

$$Lbd = \sum_{j=[1..k]} p_j \quad (2)$$

The value of H_{acc} varies according to the diversity of the column configurations: common (repeated) configurations in the *Attribute-Record* table (representing duplicated records with the same values of the data attributes) will lower the entropy H_{acc} , while diversity of the records will increase H_{acc} . $H_{acc} = 0$ when all records in MDS contain all values of all data attributes. That is, $k = 1$ and $p_1 = Lbd$. The values of H_{acc} for a given DS vary between 0 and H_{max} calculated for a specific MDS, where H_{max} represents the maximum entropy for that MDS when all records are different and thus there is no common information within MDS. $H_{max} = \log_2(Lbd)$ when all records in MDS are distinct, corresponding to the best-case scenario where there is no need to merge records:

$$Lbd = k \text{ and } p_j = 1, \forall j=[1..k] \quad (3)$$

The unit of measurement is information bit.

Derived Measure Acc . In order to measure accuracy independently of the volume of the MDS, we propose to normalize the entropy H_{acc} measure with H_{max} . Hence, the measurement function for the Accuracy indicator is:

$$Acc(MDS) = H_{acc} / H_{max} \quad (4)$$

$Acc(MDS)$ normalizes entropy by the best-case scenario H_{max} , thus normalization will allow data users to objectively compare

different DS within the MDS in terms of their common information. Acc value is a number between 0 and 1, 0 meaning the worst case (all data elements are common for all records), and 1 corresponding to the best-case scenario.

4.3 Theoretical Validation of Accuracy Measures

The Accuracy measures are assessed in this section with respect to the Tracking and Consistency criteria introduced in section 2.3. To illustrate the Acc indicator, we use an example of MDS at different time frames T1 and T2. MDS_{T1} shown in Figure 2, is a representation of data at T1 that will be used as an example of real-life Big Data. This will also be used to show how we can measure data elements in Big Data. For the purposes of theoretical validation, we present a modified case of MDS_{T1} where new records were added at time T2 (MDS_{T2} , see Figure 4), $T2 > T1$. In both cases, there are no duplicated records in the multiple datasets MDS_{T1} or MDS_{T2} . All records were mapped to *Attribute-Record* tables similar to the method described in [22] and the entropy was calculated as described in section 4.2.

	Dataset 1			Dataset 2			Dataset 3		
	Name	Salary	Debt	Name	Salary	Debt	Name	Salary	Debt
T1	Jill	50,000	10,000	Jessica	55,000	10,000	Chris	50,000	90,000
	Eve	40,000	0	Jenella	40,000	400	Beth	40,000	100
	Adam	75,000	5,000	Melvin	15,000	3,000	Ela	7,000	0

Figure 2: Big Dataset Illustration of Accuracy at T1

Intuitively, we expect the value of Accuracy for both multiple datasets MDS_{T1} and MDS_{T2} to correspond to the best-case scenario of maximum accuracy, where there is no need to merge records. From our intuitive understanding, we expect the entropy H_{accT2} of the DS depicted in Figure 3 to be higher than H_{accT1} . We also expect the values of the based measure H_{max} for MDS_{T2} to be higher than the corresponding values in MDS_{T1} due to the increased size of the DS at time T2.

The values of the variables Lbd , k and p at time T1 are:

$Lbd = 9$, $k = 9$, and $p_i = 1 \forall j=[1..k]$. The entropy value at time T1 for the DS depicted in Figure 3 is: $H_{accT1} = 3.1699$

At time T2 the values of the variables are: $Lbd = 18$, $k = 14$, $p_i = 2$ for $i \in \{1, 6, 12, 13\}$ and $p_i = 1$ for the remaining column configurations.

The entropy value at time T2 for the DS depicted in Figure 3 is: $H_{accT2} = 4.1699$. As expected, $H_{accT2} > H_{accT1}$. Similarly $H_{maxT2} = \log_2(18) > H_{maxT1} = \log_2(9)$

As expected, the value of the Acc measure indicates maximum Accuracy result ($Acc(MDS) = 100\%$) for both MDS_{T1} and MDS_{T2} .

	Dataset 1			Dataset 2			Dataset 3		
	Name	Salary	Debt	Name	Salary	Debt	Name	Salary	Debt
T1	Jill	50,000	10,000	Jessica	55,000	10,000	Chris	50,000	90,000
	Eve	40,000	0	Jenella	40,000	400	Beth	40,000	100
	Adam	75,000	5,000	Melvin	15,000	3,000	Ela	7,000	0
	Jacky	55,000	95,000	Robin	50,000	3,000	Ace	70,000	30,000
T2	Brook	45,000	1,500	Luffy	400	100	Roger	50,000	5,100
	Cathy	4,000	90	Zoro	0	10,000	Odin	70,000	0

Figure 3: Big Dataset Illustration of Accuracy at T2

To validate Tracking and Consistency criteria of the Accuracy measures on DS with duplicated records, we modify the MDS

shown in Figure 2 (MDS_{T1}) and Figure 3 (MDS_{T2}) as depicted in Figure 4 below:

	Dataset 1			Dataset 2			Dataset 3		
	Name	Salary	Debt	Name	Salary	Debt	Name	Salary	Debt
T1	Jill	50,000	10,000	Jessica	55,000	10,000	Chris	50,000	90,000
	Jill	50,000	10,000	Jessica	55,000	10,000	Beth	40,000	100
	Adam	75,000	5,000	Melvin	80,000	400	Beth	40,000	100
	Jacky	55,000	10,000	Robin	15,000	3,000	Ace	70,000	30,000
T2	Brook	45,000	1,500	Luffy	50,000	90,000	Ace	70,000	30,000
	Cathy	4,000	90	Zoro	400	10,000	Jenella	6,000	0

Figure 4: Illustration of Accuracy Measurement with duplicated records (T1 & T2)

Given that there are duplicated records in Figure 4 at both time T1 and time T2 MDS, intuitively we would expect the Accuracy of MDS_{T2} to be higher than the Accuracy of MDS_{T1} due to the relatively lower number of duplicated records. Our intuition would also expect not only $H_{accT2} > H_{accT1}$, but also $H_{accT1} > H_{accT2}$ and $H_{accT2} > H_{accT1}$. The above intuitive expectations are confirmed by the measurement results, where $H_{accT1} = 2.7254$ and $H_{accT2} = 3.7254$.

The value of Acc measure is calculated using the formula (4):

$Acc(MDS_{T1}) = H_{accT1} / H_{maxT1}$, where $H_{maxT1} = 3.17$, thus $Acc(MDS_{T1}) = 86.97\%$. Similarly, $Acc(MDS_{T2}) = 89.34\%$

Based on the analysis of the above measurement results we can conclude that both Tacking, and Consistency criteria hold for the Accuracy measures, thus we proved their theoretical validity.

4.4 Accuracy Indicator for MDS

We propose to visualize the Accuracy indicator results by depicting the Acc values of MDS graphically; this will allow data engineers to easily trace the accuracy of individual DS and identify those MDS whose records need to be analyzed further and merged, where applicable. Figure 5 illustrates the Accuracy Profile graph for the four MDS (MDS_{T1} , MDS_{T2} , MDS_{T1} and MDS_{T2}).

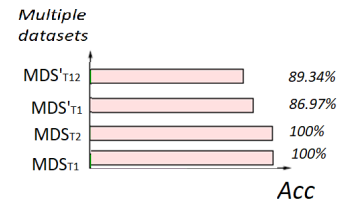


Figure 5: Illustration of the Accuracy Profile Graph for MDS

Hence, Acc indicator of Validity not only allows to objectively compare different MDS in terms of their accuracy, but likewise visualizes the Accuracy measurement results to facilitate the decision-making of the Big Data users.

5 Credibility Indicator (Cre)

5.1 Notion of Credibility

The notion of credibility in ISO/IEC DIS 25024 standard represents “the degree to which data has attributes that are true and accepted

by users in a specific context of use” [20]. In order to measure credibility, we assume the existence of up-to-date information on qualified sources.

5.2 Base Measures and Derived Measures

Base Measures Nds and Nds_{cr} . Let $cre_{source}: DS \rightarrow [0...1]$ be a function that returns 1 if the source of a DS is qualified for use, or 0 otherwise. Two base measures are defined in the measurement model of credibility:

- *Number of DS in Big Data set (Nds).* Nds is a simple counting of the total number of DS in MDS.
- *Number of credible DS in Big Data set (Nds_{cr}),* where the measurement method is counting of DS with qualified sources:

$$Nds_{cr}(MDS) = \sum_{DS \in MDS} cre_{source}(DS) \quad (5)$$

Derived measures Cre . We define Credibility measure as a ratio of the total number of credible DS and all DS. The measurement function for Cre is specified as follows:

$$Cre(MDS) = Nds_{cr}(MDS)/Nds(MDS) \quad (6)$$

Regular collection of Cre measurement data would allow practitioners to gain timely valuable control over the credibility of the data sources in their MDS and eventually trace the MDS credibility over time.

5.3 Illustration of the Cre Indicator

We illustrate the Cre indicator through a simple extract of MDS at two-time frames. The first base measure we need to calculate is Nds , which is defined as the number of DS present in MDS:

$Nds = 3$. Next, we assess the credibility of the DS sources, where $cre_{source}(DS)$ is set to 0 or 1 value, depending on whether or not a specific DS is credible. In this example we assume that DS_1 and DS_3 are credible: $cre_{source}(DS_1) = cre_{source}(DS_3) = 1$ and

$cre_{source}(DS_2) = 0$. We also assume that the credibility of the DS at times T1 and T2 remain the same. Next, we normalize the credibility of MDS by Nds . The measurement value of $Cre(MDS)$ is $\frac{2}{3}$ (or 66%), which indicated the proportion of credible DS.

	Dataset 1			Dataset 2			Dataset 3		
	Name	Salary	Debt	Name	Salary	Debt	Name	Salary	Debt
T1	Jill	50,000	10,000	Jessica	55,000	10,000	Chris	50,000	90,000
	Jill	50,000	10,000	Jessica	55,000	10,000	Beth	40,000	100
	Adam	75,000	5,000	Melvin	80,000	400	Beth	40,000	100
T2	Jacky	55,000	10,000	Robin	15,000	3,000	Ace	70,000	30,000
	Brook	45,000	1,500	Luffy	50,000	90,000	Ace	70,000	30,000
	Cathy	4,000	90	Zoro	400	10,000	Jenella	6,000	0

Figure 6: Illustration of Credibility with Duplicated Records (Time T1 and T2)

5.4 Theoretical Validation of Credibility

In this section we assess the Tracking and Consistency criteria of the Cre measures. Intuitively, the more credible sources that exist in MDS, the higher the value of credibility in MDS. In order the

validate the Cre measurement values against this intuitive expectation, we fix the value of Nds through time and track the changes of $cre_{source}(DS)$ data. In the previous example (see section 5.3) the value of $cre_{source}(DS)$ doesn’t change from T1 to T2, neither does $cre(MDS)$, as expected. If we, however, assume that at time T2 the credibility of DS_3 changes ($cre_{source}(DS_3) = 1$ at time T2), then we expect the credibility of the MDS_{T2} to increase. The measurement value of $cre(MDS_{T2})$ proves that the intuitive expectation is preserved by the measurement value of Credibility indicators, which increased from 0.66 to 1 (meaning 100% credible MDS).

These calculations establish the theoretical validity of the Credibility measures, as required by the representational theory of measurement.

6 Compliance Indicators (rec_Comp , DS_comp , MDS_comp)

In this section we introduce a hierarchy of measures for compliance, which evaluate compliance at the record (REC), dataset (DS) and multiple datasets (MDS) levels reflecting the corresponding entities in the NIST hierarchy.

6.1 Notion of Compliance

Compliance is defined as the degree to which data has attributes that adhere to standards, conventions or regulations in force and similar rules relating to data quality in specific context of use, according to ISO/IEC 25024 definition. This means that whether or not a data element is deemed as compliant depends on the judgment of the data scientists, the organization, standards and local laws and regulations.

6.2 Base Measures and Derived Measures

We define a function $Comp_{source}: Rec \rightarrow [0...1]$ that returns 1 if the source record is compliant with the set of standards that have been set by the researchers; otherwise, the returned value is 0.

Base Measure rec_comp . The base measure rec_comp counts the number of compliant records in a DS, as defined below:

$$rec_comp(DS) = \sum_{rec \in DS} Comp_{source}(rec) \quad (7)$$

Derived Measures DS_Comp and MDS_Comp . The proposed derived measure for Compliance is defined as a ratio of the Big Data entities (records, DS, or MDS) that have values and/or format that conform to standards, conventions or regulations, divided by the total number of data entities. We propose two derived measures for the Compliance indicator measuring the above ratio at the level of a DS and the level of MDS as defined below:

DataSet Compliance DS_comp . The measurement function for Compliance along a DS is defined by $DS_comp(DS)$ as follows,

$$DS_comp(DS) = \frac{rec_comp(DS)}{Ldst(DS)} \quad (8)$$

where $Ldst(DS)$ is the number of records in a specific DS.

Multiple DataSets Compliance MDS_Comp (MDS). Finally, we define a measurement function for quantifying objectively Compliance across all DS in MDS as follows:

$$MDS_Comp(MDS) = \frac{\sum_{DS \in MDS} N_{rec_Comp}(DS)}{N_{ds}(MDS)} \quad (9)$$

Regular collection of Compliance measurement data would be necessary to flag data that does not comply with local laws and regulations. For instance, in the Healthcare industry, HIPAA in the USA or PIPEDA in Canada require compliance with their privacy and data security regulations by law, thus the measurement of compliance for such sensitive data will become imperative.

6.3 Illustration of the Compliance Indicators

Figure 6 shows the same data as in Figure 7, where the data that is non-compliant is highlighted. We assume that for this example, all data elements in Salary or Debt columns must be numerical (commas are allowed). In this example, $rec_Comp(DS_1) = 2$, $rec_Comp(DS_2) = 1$ and $rec_Comp(DS_3) = 3$ at time T1. The values of the Compliance Indicator at the DS level at time T1 are as follows: $DS_Comp(DS_1) = 0.66$, $DS_Comp(DS_2) = 0.33$ and $DS_Comp(DS_3) = 1$. The result of the Compliance at time T1 in terms of MDS is defined to be the average compliance of all DS; $MDS_Comp(MDS_{T1}) = 0.66$. We perform the same steps to measure Compliance at record, DS and MDS levels in Time T2. Finally, $MDS_Comp(MDS_{T2}) = 0.72$. As expected, $MDS_Comp(MDS_{T2}) > MDS_Comp(MDS_{T1}) = 0.66$

	Dataset 1			Dataset 2			Dataset 3		
	Name	Salary	Debt	Name	Salary	Debt	Name	Salary	Debt
T1	Jill	50,000	10,000	Jessica	55,000	10000\$	Chris	50,000	90,000
	Jill	50,000	10,000	Jessica	55,000	10000\$	Beth	40,000	100
	Adam	75,000	5000\$	Melvin	80,000	400	Beth	40,000	100
	Jacky	55,000	10,000	Robin	15,000	3,000	Ace	70,000	30,000
T2	Brook	45,000	1,500	Luffy	50,000	90000\$	Ace	70,000	30,000
	Cathy	4,000	90\$	Zoro	400	10,000	Jenella	6,000	0

Figure 7. Illustration of non-compliant data in MDS

6.4 Theoretical Validation of the Compliance Measures

Based on the meaning of Compliance, the more credible datasets the Big Data contains, the larger the Cre indicator value. The perception of ‘more’ should be preserved in the mathematics of the measure: the more compliant records that exist in a particular dataset, the higher the rate of compliance as defined earlier. We validate theoretically Compliance by fixing the value of Nds (MDS) to 3 and tracking the change in MDS_Comp (MDS) through time: We see from the example above that, as the value of S_Comp (DS₁) and DS_Comp (DS₂) at the DS level increases from T1 to T2, Compliance at the MDS level increases from 0.66 to 0.72. which represents a 9% increase. This is to be expected: the percent change of records compliance in the DS increases respectively by of 8.5% (DS₁) and 9% (DS₂).

The above calculations establish the theoretical validity of the Compliance measures by demonstrating both the Tracking and Consistency criteria, as required by the representational theory of measurement.

6.5 Hierarchy of the Validity Measures

The objective of this section is to define the Validity indicator Mval that would allow to objectively compare different Big Data sets in terms of the indicators Accuracy, Credibility and Compliance, and to present graphically this new hierarchical measurement model tailored specifically to the Validity of Big Data.

6.6 Validity Indicator Mval

Validity Indicator proposed in this research is defined as a vector $Mval = (Acc(MDS), Cre(MDS), DS_comp(MDS), MDS_comp(MDS))$

that reflects correspondingly the accuracy, credibility and compliance of the underlying data at the level of DS or MDS.

7 The Measurement Hierarchy

The measurement information model proposed in this work is a hierarchical structure linking the goal of Big Data Validity to the relevant entities and attributes of concern, such as an entropy of a MDS, number of records, number of DS, etc. In our approach, the Validity characteristics were decomposed through three layers as depicted in Figure 8.

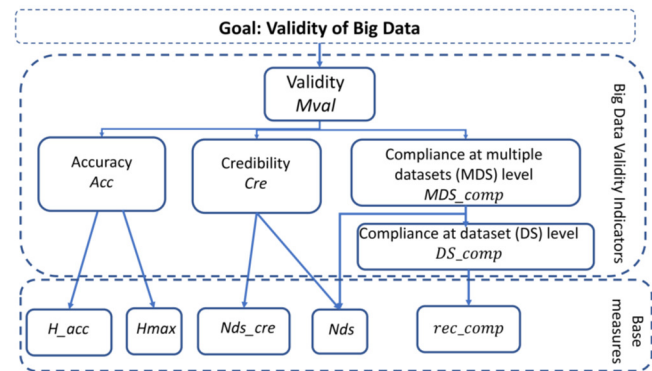


Figure 8: Hierarchical Measurement Model of Validity

The measurement information model defines how the relevant attributes are quantified and converted to indicators that provide a basis for decision-making.

8 Conclusions and Future Work

In this paper, we proposed a new theoretically valid measurement information model to evaluate Validity of Big Data in the context of the MEGA framework, applicable to a variety of existing Big Data Pipelines [9]. Four levels of entities have been considered in the definitions of the measures, as derived from the NIST hierarchy: data element, record, DS, and MDS. The model elements are compliant with ISO/IEC/IEEE Std. 15939 guidelines for their definitions, where five base measures are first defined, assembled into four derived measures, evolving into four indicators. The model is suitable for Big Data in any forms of structured, unstructured and semi-structured data. Theoretical validation of the Validity measures has been demonstrated.

We illustrated the Validity measurement model by collecting measurement data on small examples and showed how the Validity indicators Accuracy, Credibility and Compliance can be used to monitor data quality issues that may arise. The relevance of such model for the industry can be illustrated with three simple examples of usage of these measures and indicators:

- Accuracy (Acc) and its profile: Accuracy is useful to objectively compare MDS in terms of their information content, as well as to oversight variations of Big Data Accuracy over time. A decrease of Accuracy might trigger investigation as actions might be needed to merge duplicated (common) information.
- Credibility (Cre) and its trend allows easy and objective comparisons of MDS in terms of their credibility. A Credibility trend showing a decrease might trigger investigation as a source of data could be damaged or unavailable.
- Compliance (Comp) at DS and MDS levels, and the corresponding trends: Compliance allows to track an important characteristic of Validity at the level of record, DS and MDS; decrease in the measurement results over time might trigger investigation of the potential legal issues with the usage of the Big Data.

Our future research will enhance the theoretical findings presented in this paper with empirical evidence through evaluation of these measures with open-access data and industry data.

REFERENCES

- [1] R. Walker, From big data to big profits: Success with data and analytics. Oxford University Press, 2015.
- [2] J. Lee, et al. Recent advances and trends in predictive manufacturing systems in big data environment. *Manufacturing letters* 1.1 (2013): 38–41.
- [3] J. Andreu-Perez, C. C. Y. Poon, R. D. Merrifield, S. T. C. Wong and G. Yang, *Big Data for Health*, in *IEEE Journal of Biomedical and Health Informatics*, vol. 19, no. 4, pp. 1193–1208, July 2015, doi: 10.1109/JBHI.2015.2450362.
- [4] L. Ramaswamy, V. Lawson and S. V. Gogineni, Towards a quality-centric big data architecture for federated sensor services, *Proc. IEEE Int. Congr. Big Data*, pp. 86–93, Jun./Jul. 2013.
- [5] B. Kelly and I. Knezevic, Big Data in food and agriculture. *Big Data & Society* 3.1 (2016): 2053951716648174.
- [6] D. W. Bates, S. Saria, L. Ohno-Machado, A. Shah and G. Escobar, Big data in healthcare: Using analytics to identify and manage high-risk and high-cost patients, *Health Affairs*, vol. 33, pp. 1123–1131, 2014.
- [7] Yu Shui, and Song Guo, eds. Big data concepts, theories, and applications. Springer, 2016.
- [8] V. Gudivada, A. Apon, J. Ding. Data quality considerations for big data and machine learning: Going beyond data cleaning and transformations. *International Journal on Advances in Software* 10.1 (2017): 1–20.
- [9] D. Bhardwaj, O. Ormandjieva. Toward a Novel Measurement Framework for Big Data (MEGA). *Proc. 3rd IEEE International Workshop on Big Data Computation, Analysis & Applications (BDCAA 2021), COMPSAC '21 Workshop*.
- [10] O. Ormandjieva, M. Omidbakhsh, S. Trudel. Measuring the 3V's of Big Data: A Rigorous Approach. *Proc. IWSM-MENSURA 2020: In Proc. Joint conference of the 30th International Workshop on Software Measurement (IWSM) and the 15th International Conference on Software Process and Product Measurement (MENSURA)*, Mexico City, Mexico, Oct. 2020.
- [11] O. Ormandjieva, M. Omidbakhsh, S. Trudel. Measurement Model for Veracity of Big Data. *The 7th International Symposium on Big Data Principles, Architectures & Applications (BDAA 2020)*. As part of 18th International Conference on High Performance Computing & Simulation (HPCS 2020). In *Proc. HPCS 2020*, Jan. 2021.
- [12] NIST U.S. Department of Commerce, Special Publication 1500-1, NIST Big Data Interoperability Framework: Volume 1, Definitions. Volume2, Big Data Taxonomies (2018). NIST Big Data Interoperability Framework. Volume 1, definitions. In: Tech. rep. NIST SP 1500-1. National Institute of Standards and Technology; 2015. <https://doi.org/10.6028/NIST.SP.1500-1>.
- [13] ISO/IEC/IEEE 15939, 2017-04, Systems and Software Engineering - Measurement Process (2017).
- [14] N. Fenton, J. Bieman. Software Metrics: A Rigorous and Practical Approach, 3rd edn. CRC Press. <https://doi.org/10.1201/b1746> (2014).
- [15] IEEE Std 1061. IEEE Standard for Software Quality Metrics Methodology (1998).
- [16] I. Taleb, R. Dssouli and M. A. Serhani, "Big data pre-processing: A quality framework", *Proc. IEEE Int. Congr. Big Data*, pp. 191–198, Jun./Jul. 2015.
- [17] P. Pääkkönen, D. Pakkala, Reference architecture and classification of technologies, products and services for big data systems, *Big Data Res.*, Jan. 2015. DOI: 10.1016/j.bdr.2015.01.001
- [18] A. Immonen, P. Pääkkönen, E. Ovaska. Evaluating the quality of social media data in big data architecture. *IEEE Access*, 3, 2028–2043 (2015).
- [19] J. Merino, I. Caballero, B. Rivas, M. Serrano & M. Piattini, A data quality in use model for big data. *Future Generation Computer Systems*, 63, 123–130, 2017.
- [20] ISO/IEC DIS 25024, JTC1/SC7/WG6 N762, 2015-07-16, Systems and Software Engineering- Systems and Software Quality Requirements and Evaluation (SQuaRE)-Measurement of data qual
- [21] N. Zhou, G. Huang & S. Zhong (2018). Big data validity evaluation based on MMTD. *Mathematical Problems in Engineering*, 6 pages, 2018.
- [22] V. Emden, An analysis of complexity. *Mathematisch Centrum Amsterdam*. 1971.

Viral pneumonia images classification by Multiple Instance Learning: preliminary results

Matteo Avolio

matteo.avolio@unical.it

Department of Mathematics and Computer Science,
University of Calabria
Italy

Eugenio Vocaturo

e.vocaturo@dimes.unical.it

DIMES, University of Calabria
Italy
CNR NanoTec, Cosenza
Italy

Antonio Fuduli

antonio.fuduli@unical.it

Department of Mathematics and Computer Science,
University of Calabria
Italy

Ester Zumpano

e.zumpano@dimes.unical.it

DIMES, University of Calabria
Italy

ABSTRACT

At the end of 2019, the World Health Organization (WHO) referred that the Public Health Commission of Hubei Province, China, reported cases of severe and unknown pneumonia, characterized by fever, malaise, dry cough, dyspnoea and respiratory failure, which occurred in the urban area of Wuhan. A new coronavirus, SARS-CoV-2, was identified as responsible for the lung infection, now called COVID-19 (coronavirus disease 2019). Since then there has been an exponential growth of infections and at the beginning of March 2020 the WHO declared the epidemic a global emergency. An early diagnosis of those carrying the virus becomes crucial to contain the spread, morbidity and mortality of the pandemic. The definitive diagnosis is made through specific tests, among which imaging tests play an important role in the care path of the patient with suspected or confirmed COVID-19. Patients with serious COVID-19 typically experience viral pneumonia.

In this paper we launch the idea to use the Multiple Instance Learning paradigm to classify pneumonia X-ray images, considering three different classes: radiographies of healthy people, radiographies of people with bacterial pneumonia and of people with viral pneumonia. The proposed algorithms, which are very fast in practice, appear promising especially if we take into account that no preprocessing technique has been used.

CCS CONCEPTS

• **Applied computing** → **Health informatics**; • **Computing methodologies** → **Machine learning**.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Association for Computing Machinery.

ACM ISBN 978-1-4503-8991-4/21/07.

<https://doi.org/10.1145/3472163.3472170>

KEYWORDS

Multiple Instance Learning, Viral pneumonia, Image classification.

ACM Reference Format:

Matteo Avolio, Antonio Fuduli, Eugenio Vocaturo, and Ester Zumpano. 2021. Viral pneumonia images classification by Multiple Instance Learning: preliminary results. In *25th International Database Engineering & Applications Symposium (IDEAS 2021)*, July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 5 pages. <https://doi.org/10.1145/3472163.3472170>

1 INTRODUCTION

The world is coping with the COVID-19 pandemic. COVID-19 is caused by a Severe Acute Respiratory Syndrome Coronavirus (SARS-CoV-2) and its common symptoms are: fever, dry cough, fatigue, short breathing, vanishing of taste, loss of smell. The first known case of this novel Coronavirus disease was reported in Wuhan, China in the last days of 2019 [15] and since then, the virus propagates all over the world. On March 11, 2020 the WHO declared the epidemic a global emergency (pandemic). Lockdown measures and drastic restrictions of movements and social life are affecting the lives of billions of people. COVID-19 is the most significant global crisis since the Second World War, but its repercussions are exceeding those of a war. At the time of this writing, May 4, 2021, the total confirmed COVID-19 deaths according to WHO are 3.195.624 and this number is likely to be underestimated. The most effective way to limit the spread of COVID-19 and the number of deaths is to identify infected persons at an early stage of the disease and many different proposals have been investigated for the development of automatic screening of COVID-19 from medical images analysis.

COVID-19 has interstitial pneumonia as the predominant clinical manifestation. When we talk about the interstitium we mean a particular entity located between the alveolus and the capillaries, which is investigated mainly with radiological techniques. Radiological imaging does not represent a diagnostic criterion for Coronavirus (Sars-Cov2) infection, but it is able to highlight any pneumonia that can be associated with it, in this case it is possible to see an opacity on the radiograph, called *thickening*. The X-ray will show a greater extension of the pulmonary thickening: in the X-ray the lungs will appear, so to speak, more and more *white*.

Therefore, while huge challenges need to be faced, medical imaging analysis arises as a key factor in the screening of viral pneumonia from bacteria pneumonia. In reasoning on the assessment of COVID-19, chest radiography (CXR) and computed tomography (CT) are used. CT imaging shows high sensitivity, but X-ray imaging is cheaper, easier to perform and in addition (portable) X-ray machines are much more available also in poor and developing countries.

In [18] it is conducted a retrospective study of 338 patients (62% men; mean age 39 years) who presented to the emergency room in New York with confirmed COVID-19. They divided each patient's chest X-ray into six zones (three per lung) and generated a severity score based on the presence or absence of opacity in each zone (maximum score six, minimum score zero).

The idea underlying this work arises within the described scenario, characterized by an intense activity of scientific research, aimed at supporting fast solutions for diagnostics on COVID-19, which is a special case of viral pneumonia. Considering that there are recurring features that characterize the radiographs of patients affected by viral pneumonia, we propose a chest X-ray classification technique based on the Multiple Instance Learning (MIL) approach. We have considered a subset of images taken from the public Kaggle chest X-ray dataset [16] from which we have randomly extracted 50 images related to radiography of healthy people, 50 of people with bacterial pneumonia and 50 of people with viral pneumonia. This data set is widely used in the literature in connection with specific COVID-19 data sets, as reported in [2] (see for example [1, 12–14, 19]).

The preliminary results we present appear encouraging, especially if we consider that no preprocessing technique has been used on the X-ray images.

The paper is organized in the following way. In Section 2 we describe the MIL paradigm and the related techniques used for our preliminary experiments, presented in Section 3. Finally, in Section 4 some conclusions are drawn, accompanied by some considerations on possible future research directions.

2 MULTIPLE INSTANCE LEARNING

Multiple Instance Learning [11] is a classification technique consisting in the separation of point sets: such sets are called bags and the points inside the sets are called instances. The main difference of a MIL approach with respect to the classical supervised classification is that in the learning phase only the class labels of the bags are known, while the class labels of the instances remain unknown.

The MIL paradigm finds applications in a lot of contexts such as in text classification, bankruptcy prediction, image classification, speaker identification and so on.

A particular role played by MIL is in medical image and video analysis, as shown in [17]. Diagnostics by means of image analysis is an important field in order to support physicians to have early diagnoses. We focus on binary MIL classification with two classes of instances, on the basis of the so-called standard MIL assumption, which considers positive a bag containing at least a positive instance and negative a bag containing only negative instances. Such assumption fits very well with diagnostics by images: in fact a patient is non-healthy (i.e. is positive) if his/her medical scan (bag) contains

at least an abnormal subregion and is healthy if all the subregions forming his/her medical scan are normal.

In [7] a MIL approach has been used for melanoma detection on some clinical data constituted by color dermoscopic images, with the aim to discriminate between melanomas (positive images) and common nevi (negative images). The obtained results encourage to investigate possible use of MIL techniques also in viral pneumonia detection by means of chest X-rays images, which is the scope of this paper. In particular, using binary MIL classification techniques, our aim is to discriminate between X-rays images of healthy patients versus patients with bacteria pneumonia, healthy patients versus patients with viral pneumonia and patients with bacteria pneumonia versus patients with viral pneumonia. The results we present are preliminary since no image preprocessing technique has been adopted.

The MIL techniques we use in this work for chest X-ray chest image classification fall into the instance-level approaches and they are object of the next subsections. Both of them are designed taking into account the so-called standard MIL assumption, stating that a bag is positive if it contains at least a positive bag and it is negative otherwise.

2.1 The MIL-RL algorithm

MIL-RL algorithm [5] is an instance-level technique based on solving, by Lagrangian relaxation [10], the Support Vector Machine (SVM) type model proposed by Andrews et al. in [3]. Such model, providing an SVM separating hyperplane of the type

$$H(w, b) \triangleq \{x \in \mathbb{R}^n \mid w^T x + b = 0\}, \quad (1)$$

is the following:

$$\left\{ \begin{array}{l} \min_{y, w, b, \xi} \quad \frac{1}{2} \|w\|^2 + C \sum_{i=1}^m \sum_{j \in J_i^+} \xi_j + C \sum_{i=1}^k \sum_{j \in J_i^-} \xi_j \\ \xi_j \geq 1 - y_j(w^T x_j + b) \quad j \in J_i^+, i = 1, \dots, m \\ \xi_j \geq 1 + (w^T x_j + b) \quad j \in J_i^-, i = 1, \dots, k \\ \sum_{j \in J_i^+} \frac{y_j + 1}{2} \geq 1 \quad i = 1, \dots, m \\ y_j \in \{-1, +1\} \quad j \in J_i^+, i = 1, \dots, m \\ \xi_j \geq 0 \quad j \in J_i^+, i = 1, \dots, m \\ \xi_j \geq 0 \quad j \in J_i^-, i = 1, \dots, k, \end{array} \right. \quad (2)$$

where:

- m is the number of positive bags;
- k is the number of negative bags;
- x_j is the j -th instance belonging to a bag;
- J_i^+ is the index set corresponding to the instances of the i -th positive bag;
- J_i^- is the index set corresponding to the instances of the i -th negative bag.

Variables b and w correspond respectively to the bias and normal to the hyperplane, variable ξ_j gives a measure of the misclassification error of the instance x_j , while y_j is the class label to be assigned to the instances of the positive bags. The positive parameter C tunes the weight between the maximization of the margin, obtained by minimizing the Euclidean norm of w , and the minimization of the misclassification errors of the instances. Finally, the constraints

$$\sum_{j \in J_i^+} \frac{y_j + 1}{2} \geq 1 \quad i = 1, \dots, m \quad (3)$$

impose that, for each positive bag, at least one instance should be positive (i.e. with label equal to +1).

Note that, when $m = k = 1$ and $y_j = +1$ for any j , problem (2) reduces to the classical SVM quadratic program.

The core of MIL-RL is to solve, at each iteration, the Lagrangian relaxation of problem (2), obtained by relaxing constraints (3):

$$\left\{ \begin{array}{l} \min_{y, w, b, \xi} \quad \frac{1}{2} \|w\|^2 + C \sum_{i=1}^m \sum_{j \in J_i^+} \xi_j + C \sum_{i=1}^k \sum_{j \in J_i^-} \xi_j + \sum_{i=1}^m \lambda_i \left(1 - \sum_{j \in J_i^+} \frac{y_j + 1}{2} \right) \\ \xi_j \geq 1 - y_j(w^T x_j + b) \quad j \in J_i^+, i = 1, \dots, m \\ \xi_j \geq 1 + (w^T x_j + b) \quad j \in J_i^-, i = 1, \dots, k \\ y_j \in \{-1, +1\} \quad j \in J_i^+, i = 1, \dots, m \\ \xi_j \geq 0 \quad j \in J_i^+, i = 1, \dots, m \\ \xi_j \geq 0 \quad j \in J_i^-, i = 1, \dots, k, \end{array} \right. \quad (4)$$

where $\lambda_i \geq 0$ is the i -th Lagrangian multiplier associated to the i -th constraint of the type (3). In [5] it has been shown that, considering the Lagrangian dual of the primal problem (2), in correspondence to the optimal solution there is no dual gap between the primal and dual objective functions.

2.2 The mi-SPSVM algorithm

Algorithm mi-SPSVM has been introduced in [8] and it exploits the good properties exhibited for supervised classification by the SVM technique in terms of accuracy and by the PSVM (Proximal Support Vector Machine) approach [9] in terms of efficiency. It computes a separating hyperplane of the type (1) by solving, at each iteration, the following quadratic problem:

$$\left\{ \begin{array}{l} \min_{w, b, \xi} \quad \frac{1}{2} \left\| \begin{matrix} w \\ b \end{matrix} \right\|^2 + \frac{C}{2} \sum_{j \in J^+} \xi_j^2 + C \sum_{j \in J^-} \xi_j \\ \xi_j = 1 - (w^T x_j + b) \quad j \in J^+ \\ \xi_j \geq 1 + (w^T x_j + b) \quad j \in J^- \\ \xi_j \geq 0 \quad j \in J^-, \end{array} \right. \quad (5)$$

by varying of the sets J^+ and J^- , which contain the indexes of the instances currently considered positive and negative, respectively. At the initialization step, J^+ contains the indexes of all the instances of

the positive bags, while J^- contains the indexes of all the instances of the negative bags. Once an optimal solution, say (w^*, b^*, ξ^*) , to problem (5) has been computed, the two sets J^+ and J^- are updated in the following way:

$$J^+ := J^+ \setminus \bar{J} \quad \text{and} \quad J^- := J^- \cup \bar{J}$$

where

$$\bar{J} = \{j \in J^+ \setminus J^* \mid w^{*T} x_j + b^* \leq -1\},$$

with $J^* = \{j_i^*, i = 1, \dots, m \mid w^{*T} x_{j_i^*} + b^* \leq -1\}$ and

$$j_i^* \triangleq \arg \max_{j \in (J_i^+ \cap J^+)} \{w^{*T} x_j + b^*\}.$$

Some comments on the updating of the sets J^+ and J^- are in order. A particular role in the definition of the set \bar{J} is played by the set J^* , introduced for taking into account constraints (3). We recall that such constraints impose the satisfaction of the standard MIL assumption, stating that, for each positive bag, at least one instance must be positive. At the current iteration, the set J^* is the index set (subset of J^+) corresponding to the instances closest, for each positive bag, to the current hyperplane $H(w^*, b^*)$ and strictly lying in the negative side with respect to it. If an index, say $j_i^* \in J^*$, corresponding to one of such instances entered the set J^- , all the instances of the i -th positive bag would be considered negative by problem (5), favouring the violation of the standard MIL assumption. This is the reason why the indexes of J^* are prevented from entering the set J^- : in this way, for each positive bag, at least an index corresponding to one of its instances is guaranteed to be inside J^+ .

3 NUMERICAL RESULTS

Algorithms MIL-RL and mi-SPSVM have been preliminary tested on 150 X-ray chest images, randomly taken from the public dataset [16] available at <https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia>: 50 images are of healthy people (Figure 1), 50 correspond to people with bacterial pneumonia (Figure 2) and 50 are of people with viral pneumonia (Figure 3).

We have used the same Matlab implementation of MIL-RL as in [7] and the same Matlab implementation of mi-SPSVM tested in [8].

As for the segmentation process, we have adopted a procedure similar to that one used in [6]. In particular, we have reduced the resolution of each image to 128×128 pixels dimension and we have grouped the pixels in appropriate square subregions (blobs). In this way, each image is represented as a bag, while a blob corresponds to an instance of the bag. For each instance (blob), we have considered the following 10 features:

- the average and the variance of the grey-scale intensity of the blob: 2 features;
- the differences between the average of the grey-scale intensity of the blob and that ones of the adjacent blobs (upper, lower, left, right): 4 features;
- the differences between the variance of the grey-scale intensity of the blob and that ones of the adjacent blobs (upper, lower, left, right): 4 features.

Since our approach is of the binary type, we have performed the following X-ray chest images classification:

- bacterial pneumonia (positive) images versus normal (negative) images (Table 1);

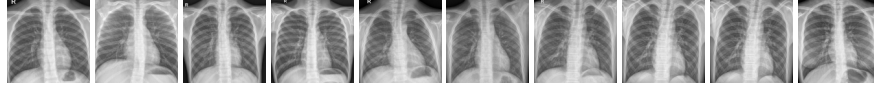


Figure 1: Examples of chest X-ray images of healthy people

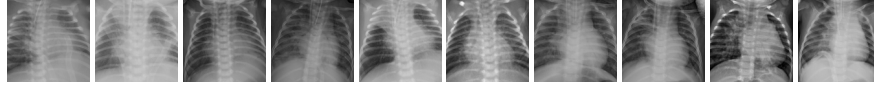


Figure 2: Examples of chest X-ray images of people with bacterial pneumonia

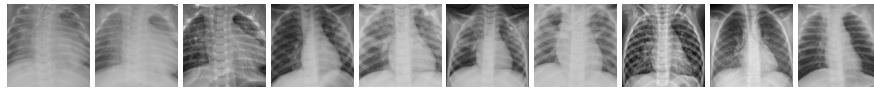


Figure 3: Examples of chest X-ray images of people with viral pneumonia

- viral pneumonia (positive) images versus normal (negative) images (Table 2);
- viral pneumonia (positive) images versus bacterial pneumonia (negative) images (Table 3).

In particular, in order to consider different sizes of the testing and training sets, we have used three different validation protocols: the 5-fold cross-validation (5-CV), the 10-fold cross-validation (10-CV) and the Leave-One-Out validation. As for the optimal computation of the tuning parameter C characterizing the models (2) and (5), in both the cases we have adopted a bi-level approach of the type used in [4] and in [7].

In Tables 1, 2 and 3 we report the average values provided by MIL-RL and mi-SPSVM in terms of correctness (accuracy), sensitivity, specificity and F-score, computed on the testing set. We also report the average CPU time spent by the classifier to determine the separation hyperplane.

We observe that mi-SPSVM is clearly faster than MIL-RL and, in general, it classifies better, even if the accuracy results provided by the two codes appear comparable. In classifying bacterial pneumonia (Table 1) and viral pneumonia (Table 2) against normal X-ray chest images we obtain high values of accuracy (about 90%) and sensitivity (about 94%). We recall that the sensitivity (also called true positive rate) is a very important parameter in diagnostics since it measures the proportion of positive patients correctly identified.

On the other hand, when we discriminate between the viral pneumonia and the bacterial pneumonia images (Table 3), we obtain lower results with respect to those ones reported in Tables 1 and 2, as expected since the two classes are very similar. Nevertheless, these values appear reasonable, especially in terms of sensitivity (82% provided by mi-SPSVM) and of F-score (75.93%), if we consider that no preprocessing procedure has been applied to the images.

4 CONCLUSIONS AND FUTURE WORK

In this work we have presented some preliminary numerical results obtained from classification of viral pneumonia against bacterial pneumonia and normal X-ray chest images, by means of Multiple Instance Learning (MIL) algorithms. These results appear promising, especially if we take into account that no preprocessing phase has been performed. Moreover our MIL techniques appear appealing also in terms of computational efficiency, since the separation hyperplane is always obtained in less than one second.

Future research could consist in appropriately preprocessing the images and in considering additional features to be exploited in the classification process, including also COVID-19 chest X-ray images. In fact the ambitious goal is to create a framework that can quickly support the diagnostics of COVID-19.

REFERENCES

- [1] B. Aksoy and O. K. M. Salman. Detection of COVID-19 disease in chest X-ray images with capsule networks: application with cloud computing. *Journal of Experimental and Theoretical Artificial Intelligence*, 33(3):527–541, 2021.
- [2] H. S. Alghamdi, G. Amoudi, S. Elhag, K. Saeedi, and J. Nasser. Deep learning approaches for detecting COVID-19 from chest X-ray images: A survey. *IEEE Access*, 9:20235–20254, 2021.
- [3] S. Andrews, I. Tschantzaris, and T. Hofmann. Support vector machines for multiple-instance learning. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems*, pages 561–568. MIT Press, Cambridge, 2003.
- [4] A. Astorino and A. Fuduli. The proximal trajectory algorithm in SVM cross validation. *IEEE Transactions on Neural Networks and Learning Systems*, 27(5):966–977, 2016.
- [5] A. Astorino, A. Fuduli, and M. Gaudioso. A Lagrangian relaxation approach for binary multiple instance classification. *IEEE Transactions on Neural Networks and Learning Systems*, 30(9):2662 – 2671, 2019.
- [6] A. Astorino, A. Fuduli, P. Veltri, and E. Vocaturo. On a recent algorithm for multiple instance learning. preliminary applications in image classification. In *2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 1615–1619, 2017.
- [7] A. Astorino, A. Fuduli, P. Veltri, and E. Vocaturo. Melanoma detection by means of multiple instance learning. *Interdisciplinary Sciences: Computational Life Sciences*, 12(1):24–31, 2020.
- [8] M. Avolio and A. Fuduli. A semiproximal support vector machine approach for binary multiple instance learning. *IEEE Transactions on Neural Networks and*

	5-CV		10-CV		Leave-One-Out	
	MIL-RL	mi-SPSVM	MIL-RL	mi-SPSVM	MIL-RL	mi-SPSVM
Correctness (%)	88.00	<u>91.00</u>	89.00	<u>91.00</u>	90.00	<u>91.00</u>
Sensitivity (%)	<u>94.70</u>	<u>94.70</u>	93.83	93.83	94.00	<u>94.00</u>
Specificity (%)	82.50	87.50	83.38	86.81	86.00	<u>88.00</u>
F-score (%)	88.73	<u>91.68</u>	88.89	90.74	90.38	<u>91.26</u>
CPU time (secs)	0.24	<u>0.04</u>	0.32	0.06	0.59	0.14

Table 1: Data set constituted by 50 normal X-ray chest images and 50 X-ray chest images with bacterial pneumonia: average testing values

	5-CV		10-CV		Leave-One-Out	
	MIL-RL	mi-SPSVM	MIL-RL	mi-SPSVM	MIL-RL	mi-SPSVM
Correctness (%)	87.00	88.00	84.00	87.00	86.00	<u>89.00</u>
Sensitivity (%)	86.36	93.18	90.71	90.71	88.00	<u>94.00</u>
Specificity (%)	<u>88.61</u>	83.89	77.38	83.05	84.00	<u>84.00</u>
F-score (%)	86.53	88.63	84.85	87.36	86.27	<u>89.52</u>
CPU time (secs)	0.35	<u>0.05</u>	0.56	0.06	0.58	0.07

Table 2: Data set constituted by 50 normal X-ray chest images and 50 X-ray chest images with viral pneumonia: average testing values

	5-CV		10-CV		Leave-One-Out	
	MIL-RL	mi-SPSVM	MIL-RL	mi-SPSVM	MIL-RL	mi-SPSVM
Correctness (%)	72.00	67.00	<u>75.00</u>	74.00	71.00	74.00
Sensitivity (%)	71.74	80.23	73.98	80.83	68.00	<u>82.00</u>
Specificity (%)	70.83	54.44	<u>74.86</u>	63.10	74.00	66.00
F-score (%)	70.46	69.99	<u>73.06</u>	73.69	70.10	<u>75.93</u>
CPU time (secs)	0.36	<u>0.06</u>	0.89	<u>0.06</u>	0.93	0.15

Table 3: Data set constituted by 50 X-ray chest images with bacterial pneumonia and 50 X-ray chest images with viral pneumonia: average testing values

- Learning Systems*, DOI: 10.1109/TNNLS.2020.3015442, pages 1–12, 2020. in press.
- [9] G. Fung and O. Mangasarian. Proximal support vector machine classifiers. In F. Provost and R. Srikant, editors, *Proceedings KDD-2001: Knowledge discovery and data mining*, pages 77–86. ACM, 2001.
- [10] M. Guignard. Lagrangean relaxation. *Top*, 11(2):151–200, 2003.
- [11] F. Herrera, S. Ventura, R. Bello, C. Cornelis, A. Zafra, D. Sánchez-Tarragó, and S. Vluymans. *Multiple instance learning: Foundations and algorithms*. Springer International Publishing, 2016.
- [12] D. M. Ibrahim, N. M. Elshennawy, and A. M. Sarhan. Deep-chest: Multi-classification deep learning model for diagnosing COVID-19, pneumonia, and lung cancer chest diseases. *Computers in biology and medicine*, 132, 2021.
- [13] S. Johri, M. Goyal, S. Jain, M. Baranwal, V. Kumar, and R. Upadhyay. A novel machine learning-based analytical framework for automatic detection of COVID-19 using chest X-ray images. *International Journal of Imaging Systems and Technology*, 2021. In press.
- [14] R. C. Joshi, S. Yadav, V. K. Pathak, H. S. Malhotra, H. V. S. Khokhar, A. Parihar, N. Kohli, D. Himanshu, R. K. Garg, M. L. B. Bhatt, R. Kumar, N. P. Singh, V. Sardana, R. Burget, C. Alippi, C. M. Travieso-Gonzalez, and M. K. Dutta. A deep learning-based COVID-19 automatic diagnostic framework using chest X-ray images. *Biocybernetics and Biomedical Engineering*, 41(1):239–254, 2021.
- [15] Q. Li, X. Guan, P. Wu, X. Wang, L. Zhou, Y. Tong, R. Ren, K. S. Leung, E. H. Lau, J. Y. Wong, X. Xing, N. Xiang, Y. Wu, C. Li, Q. Chen, D. Li, T. Liu, J. Zhao, M. Liu, W. Tu, C. Chen, L. Jin, R. Yang, Q. Wang, S. Zhou, R. Wang, H. Liu, Y. Luo, Y. Liu, G. Shao, H. Li, Z. Tao, Y. Yang, Z. Deng, B. Liu, Z. Ma, Y. Zhang, G. Shi, T. T. Lam, J. T. Wu, G. F. Gao, B. J. Cowling, B. Yang, G. M. Leung, and Z. Feng. Early transmission dynamics in Wuhan, China, of novel coronavirusâ€infecting pneumonia. *New England Journal of Medicine*, 382(13):1199–1207, 2020.
- [16] P. Mooney. Chest X-ray images (pneumonia). <https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia>.
- [17] G. Quellec, G. Cazuguel, B. Cochener, and M. Lamard. Multiple-instance learning for medical image and video analysis. *IEEE Reviews in Biomedical Engineering*, 10:213–234, 2017.
- [18] D. Toussie, N. Voutsinas, M. Finkelstein, M. A. Cedillo, S. Manna, S. Z. Maron, A. Jacobi, M. Chung, A. Bernheim, C. Eber, J. Concepcion, Z. A. Fayad, and Y. S. Gupta. Clinical and chest radiography features determine patient outcomes in young and middle-aged adults with COVID-19. *Radiology*, 297(1):E197–E206, 2020.
- [19] T. Zebin and S. Rezvy. COVID-19 detection and disease progression visualization: Deep learning on chest X-rays for classification and coarse localization. *Applied Intelligence*, 51(2):1010–1021, 2021.

An In-Browser Collaborative System for Functional Annotations

Yui Saeki

Dept. of Information and Computer Science,
Keio University
Yokohama, Japan
saeki@db.ics.keio.ac.jp

Motomichi Toyama

Dept. of Information and Computer Science,
Keio University
Yokohama, Japan
toyama@ics.keio.ac.jp

ABSTRACT

In this work, we used the Web Index system, which converts words into hyperlinks in arbitrary Web pages, to implement a system for sharing annotations registered on keywords within a limited group. This system allows group members to view all the written annotations by simply mousing over the keyword when it appears on a web page, facilitating sharing information and awareness in collaborative research and work. In this study, we define our system as a sticky note type annotation sharing system. In contrast to the sticky note type, our system is positioned as a functional type, but it may display annotations that are not necessary since it allows browsing on any page. To improve this point, we propose a usability judgment method that shows only the most useful posts.

CCS CONCEPTS

• **Human-centered computing** → **Collaborative and social computing systems and tools.**

KEYWORDS

Annotation, Web Index, Computer supported cooperative work

ACM Reference Format:

Yui Saeki and Motomichi Toyama. 2021. An In-Browser Collaborative System for Functional Annotations. In *25th International Database Engineering & Applications Symposium (IDEAS 2021), July 14–16, 2021, Montreal, QC, Canada*. ACM, New York, NY, USA, 7 pages. <https://doi.org/10.1145/3472163.3472275>

1 INTRODUCTION

In recent years, with the spread of the Internet, many information sharing tools have been developed and are being widely used, such as e-mail, social media, the cloud, etc. However, in order to share each piece of information in a collaborative research or work environment, it is necessary to save or transmit each piece of information to a dedicated tool or space. This makes the information sharing process very time-consuming. We thought that if information that we wanted to share appeared on a Web page, we could write a memo that could be shared directly with everyone on that word, and that would make collaborative work and research more

efficient. Therefore, in this work, we propose a system that shares the messages among a specific group by directly annotating the words in the Web page.

Various tools exist for the purpose of sharing annotations. Many of these tools register annotations on specific Web pages and documents or on words within these documents. In that case, it is difficult to share the knowledge about the word when you view the word on another page after registering the annotation for that word. Therefore, we thought that using a system that could register annotations on the words themselves and view them on any Web page would make collaborative work and research more efficient.

In this work, we call systems that register annotations only on a specific Web page or document, as is most commonly done in current tools, sticky note type annotation sharing systems. On the other hand, we propose a functional type annotation sharing system that can share annotations with words in any arbitrary page. The proposed system implements a function to annotate a specific word in a database and applies the function to words. The argument of the applied function is an arbitrary Web page. Based on this, it was positioned as a function type.

In order to implement a functional annotation sharing system, we used the Web Index system [7, 9], which is developed by Toyama laboratory, Keio University. The web Index (WIX) system generates hyperlinks from keywords in Web documents to make it easier for viewers of Web pages to access other Web pages related to words appearing in the page being viewed. The WIX system uses a WIX file that describes an entry in XML format by combining a keyword and a URL.

Functional systems can browse annotations efficiently because annotations added on a specific page can be viewed on any page. However, there are times when annotations that are not necessary or unreliable are displayed when browsing that web page. To solve this problem, we propose a method to rank the annotations and display only the top few annotations when sharing.

The structure of this paper is as follows. First, Section 2 gives an overview of the Web Index system. Section 3 describes the annotation sharing system, and Section 4 describes the evaluation. Section 5 is about improving the system. Section 6 describes related work on annotation systems and usability judgment method. Section 7 discusses conclusion and future work.

2 WEB INDEX SYSTEM

2.1 Overview

The Web Index system is a system that joins information resources on the Web. In general, current web pages have a structure in which a specific link destination is attached to a specific anchor text. On the other hand, the WIX system uses a WIX file that describes

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](https://permissions.acm.org).

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Association for Computing Machinery.

ACM ISBN 978-1-4503-8991-4/21/07.

<https://doi.org/10.1145/3472163.3472275>

a set of entries, which are pairs of keywords and URLs, in XML format. By referring to the WIX file, the WIX system convert keywords appearing in the text of any Web page into hyperlinks to the corresponding URL.

2.2 WIX File

A WIX file is a file that describes combinations of keywords and URLs in XML format. A keyword that will serve as an anchor for the links is described in a keyword element, and the URL of the corresponding Web page is described as a target element. These two are combined and called an entry. In the header element, it is also possible to describe the outline of the file, metadata, comments of the creator, etc. WIX files contain entries that share something in common, such as "List of English Wikipedia entry words", "Official professional baseball sites", and "English-Japanese dictionary". An example of a WIX file is shown below.

```
<WIX>
  <header>
    <comment>example</comment>
    <description>Apr 2020, saeki</description>
    <language>en</language>
  </header>
  <body>
    <entry>
      <keyword>Messi</keyword>
      <target>https://www.fcbarcelona.com/en/football/first-team/players/4974/lionel-messi</target>
    </entry>
    <entry>
      <keyword>Nishikori</keyword>
      <target>https://www.atptour.com/en/players/kei-nishikori/n552/overview</target>
    </entry>
  </body>
</WIX>
```

2.3 Architecture

Figure 1 shows the architecture of the Web Index system. The WIX system converts keywords received from the client into keyword hyperlinks on the server.

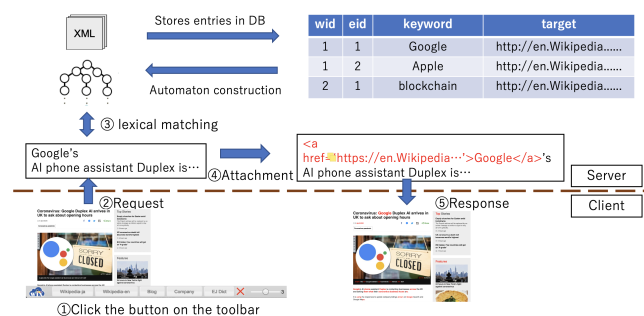


Figure 1: Architecture of the WIX system

2.4 WIX library

The WIX library stores all WIX files and manages information on a file-by-file basis. WIX file creators can upload WIX files through the WIX library, and the uploaded WIX files are decomposed into entry units and stored in the WIX DB.

2.5 Find Index

Find Index is an automaton that expands entry information in WIX DB into memory and realizes high-speed attachment. The WIX system constructs an automaton based on the Aho-Corasick method [8] and performs lexical matching.

2.6 Convert into Hyperlink

The client side of the WIX system is implemented as a Chrome extension. When this extension is installed in Chrome, a toolbar like the one shown in Figure 2 is displayed at the bottom of the browser. Clicking on the WIX filename button in this toolbar will convert the words in the web page into corresponding hyperlinks. This combining operation is called attachment. Figure 3 and Figure 4 show the web page before and after attachment using the WIX file of English Wikipedia. For example, we used a news article [10] from BBC NEWS.



Figure 2: WIX toolbar

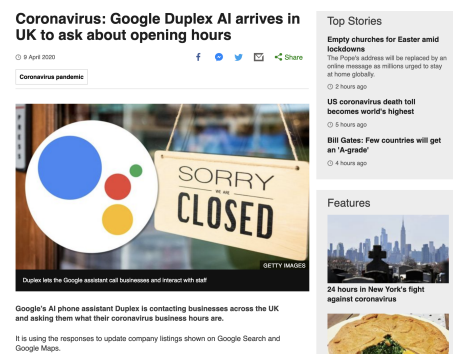


Figure 3: Before Attachment

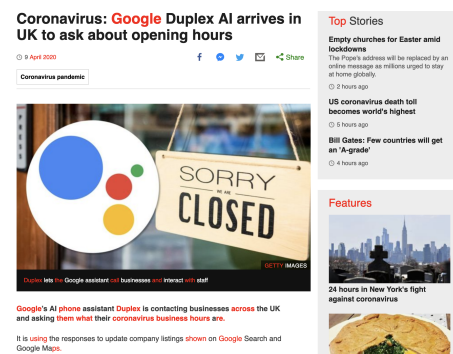


Figure 4: After Attachment



Figure 5: Toolbar of Annotation Sharing System

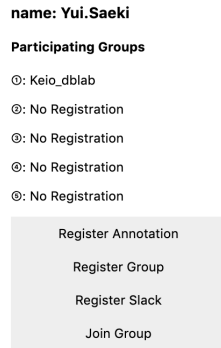


Figure 6: Popup menu of Annotation Sharing System

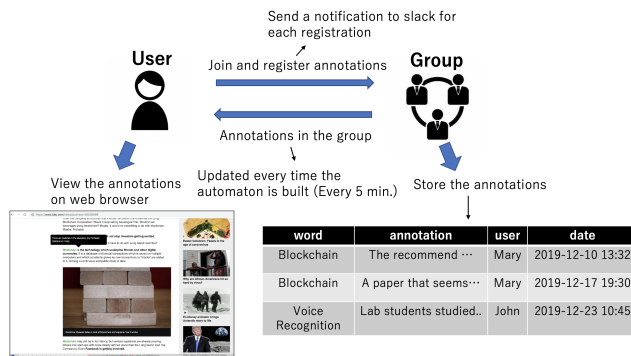


Figure 7: The Whole Picture of Annotation Sharing System

3 PROPOSED ANNOTATION SHARING SYSTEM

3.1 Overview

This proposed annotation sharing system can be used by anyone who installs it as a chrome extension. When this system is installed, a toolbar as shown in Fig. 5 is displayed at the bottom of the chrome browser, and a popup menu as shown in Fig. 6 can be used by clicking the button displayed on the right side of the address bar.

Fig. 7 shows the relationship between users and groups in this system and the whole picture of this system. Users can join groups and share annotations on words within the group. A group here is a community that can share the same annotation, and members of the group can register annotations and browse annotations in the group. Any user of this system can create new groups. Users can also send a notification of the annotation content to the registered Slack channel when the annotation is registered.

The method of performing lexicographic matching on sentences on a web page and converting the words registered in annotations is Find Index used in the WIX system. On the server side, in the

Table 1: User table

id	name
1	John
2	Mary
3	Harry

Table 2: Group table

id	name	pwd	slack
1	Database	*****	https://hooks...
2	Network	*****	https://hooks...
3	AI	*****	https://hooks...

Table 3: Join table

group_id	user_id
1	2
1	3
2	1
3	1

WIX system, keywords were converted to hyperlinks when attaching, whereas in this system, HTML was converted to HTML with annotation balloons displayed when hovering over keywords. Also, in the WIX system, the Find Index was manually constructed after newly registering the WIX file, but this system assumes that the construction will be performed every 5 minutes. If updated every 5 minutes, the annotations registered during that time cannot be immediately reflected on the Web. Therefore, in this system, a function to immediately notify the annotation registration on Slack was added.

3.2 Data management

In this section we describe how data about the users, user groups and annotations are stored on the WIX system server. Data is managed in a relational database. We describe the data schema used and illustrate it with examples.

3.2.1 User and Group DB. User and group management uses the user table, group table, and join table. The user table stores the user ID (id) and the user name (name). The join table stores the group ID (group_id) and the user ID (user_id) are stored. The group table stores the group ID (id), group name (name), password for group participation (pwd), and slack link URL (slack). Table 1, 2, 3 show examples of these tables.

These examples show that Mary and Harry are participating 'database' group and John is participating the 'network' group and 'AI' group. Users can join multiple groups.

3.2.2 Annotation DB. Annotations are managed using the annotation table and the wix_file_entry table, which is also used in the WIX system. The annotation table stores the group ID (wid), word ID (eid), user ID (uid), annotation (annotation), and registered date and time (added_time). Table 4, 5 show examples of these tables.

These examples show that 'Blockchain' has 2 annotations and 'Voice recognition' and 'Autonomous car' have an annotation. Multiple annotations can be registered for the same word.

3.3 Applying Annotations

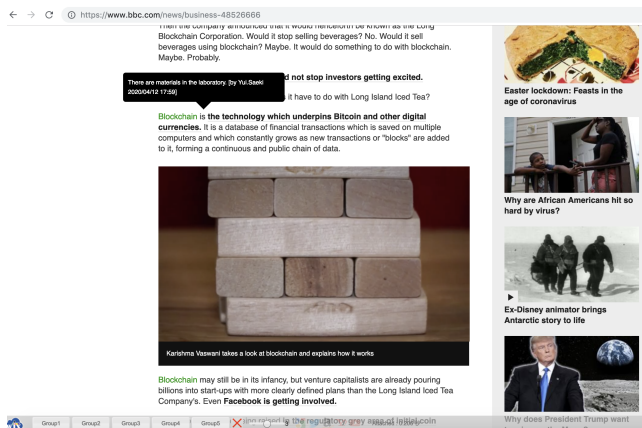
As with the Web Index system, annotation sharing is performed by clicking a button on the toolbar. Figure 8 shows an example of

Table 4: Annotation table

wid	eid	uid	annotation	added_time
2	1	2	The recommended reference is here. (https://....)	2019-12-8 13:22
2	1	2	A paper that seems to be helpful is "An In-Browser..."	2020-1-2 18:03
3	1	1	Lab students studied this subject last year.	2020-2-21 10:45
3	2	1	There are materials in our lab.	2020-2-24 19:11

Table 5: wix_file_entry table

wid	eid	keyword	target
2	1	Blockchain	annotation
3	1	Voice recognition	annotation
3	2	Autonomous car	annotation

**Figure 8: Sharing Annotation**

a web page after sharing. All the words registered in the group selected in the database turn green, and when you mouse over the word, the annotation content is displayed in a black balloon. The example of Fig. 8 used BBC NEWS [11].

4 EVALUATION

This work compared media and the target to be annotated between multiple similar tools and this proposed system.

In addition, we performed a numerical experiment and compared how annotations were shared under the assumption that this system and similar tools were used in the same environment.

4.1 Comparison with similar tools

Diigo [4], A.nnotate [2], and Adobe [1] used as similar tools for comparison. Table 6 shows the result of comparison in terms of supported formats. Table 6 shows that the proposed system has fewer supported formats than A.nnotate.

Table 6: Comparison with similar tools in terms of supported formats.

System	Supported Formats
Proposed System	HTML
Diigo	HTML
A.nnotate	PDF, Word, HTML
Adobe	PDF

Furthermore, all of these tools can share annotations instantly. Since the proposed system is functional, it has a feature that it can be applied to non-specific pages that are not included in other tools that are sticky note type.

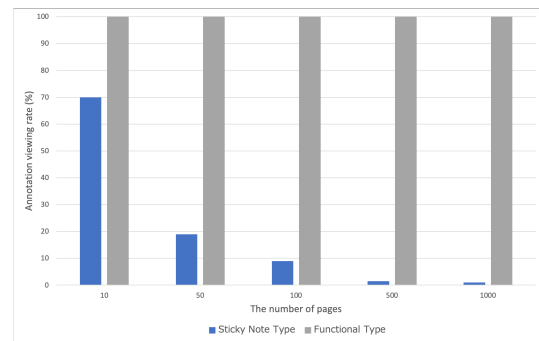
4.2 Numerical Simulation

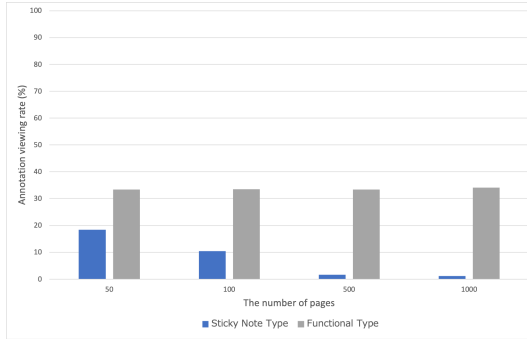
4.2.1 Setting Conditions. A numerical simulation to evaluate this functional system was performed under the following hypothetical conditions. We assume a sticky note type system that registers annotations for words on a specific Web page only, and use it as a comparison target system.

- 100 participants
- There are n pages that can be viewed (n : 10, 50, 100, 500, 1000)
- View one page per person per day (select randomly)
- Annotation is registered for m words (m : 2, 5, 25)
- Use the average number of annotation types viewed by each participant on each day.
- Calculate results on days 10, 30, and the last day.

This experiment calculate the viewing rate from the average number of annotation types viewed by each participant. The viewing rate here defines how many types of annotations among all types of annotations were viewed in each condition.

4.2.2 Result. First, a comparison based on the number of pages which have an annotation is performed. On day 10, the annotation viewing rate of the sticky type system and the functional type system when $m = 2$ was as shown in Fig. 9. The result when $m = 25$ is shown in Fig. 10. In the case of 25 types, if the number of pages is less than 25 pages, it is not possible to view all of them, so the graph excluding the case of $n = 10$ is shown.

**Figure 9: Result of m = 2, Day 10**

Figure 10: Result of $m = 25$, Day 10

From these two graphs, it can be seen that the viewing rate of the functional system is much higher than that of the sticky system in both cases. In addition, the viewing rate of the sticky type system decreases greatly with the increase in the number of pages, but the viewing rate of the functional system does not change much. Therefore, it can be inferred that the more pages, the more efficient the functional system can share annotations.

Second, we compare by period. Fig. 11 shows the viewing rate of both systems with $m = 5$ and $n = 100$. From this graph, it can be seen that the browsing rate increases as the period increases in both systems. At this time, the increase in the viewing rate was large in the sticky note system, but the viewing rate in the functional system was close to 100% on day 10, so the increase rate on day 50 was very small.

From this, when using for a long period of time, it is possible to view annotations sufficiently even with a sticky note type system, but when using for a short period of time, it is more difficult to find annotations with a sticky note type system than with a functional type system.

Finally, we compare the number of annotation words. On day 30, the viewing rate by both systems with $n = 100$ is shown in Fig. 12. From this graph, it can be seen that the viewing rate increases as the number of words increases in the sticky note type system, whereas the viewing rate decreases greatly as the number of words increases in the functional type system. This is because the number of pages on which a word exists in a sticky note type system increases, while in a functional type system, one word exists on n / m pages.

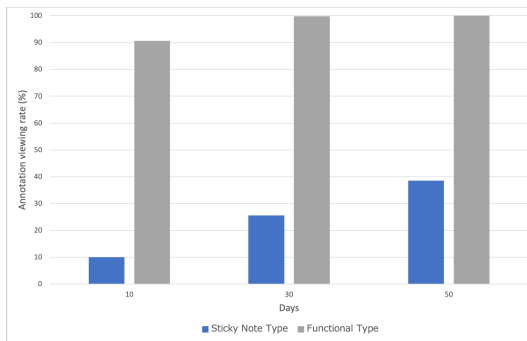
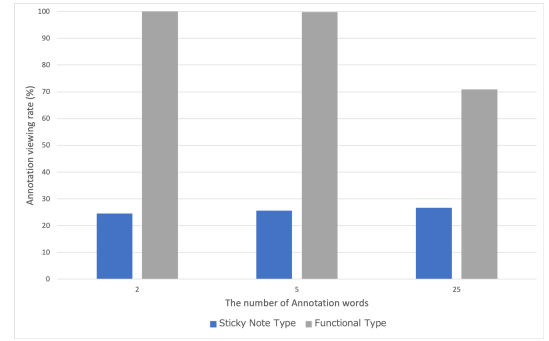
Figure 11: Result of $m = 5$, $n = 100$

Table 7: Comparison of sticky note type and functional type.

	Strength	Weakness
sticky note type	Precise communication of intent.	Fewer opportunities to browse.
functional type	Efficient sharing.	Posts that are not useful will be shown.

Figure 12: Result of $n = 100$, Day 30

From the above comparison, it was found that the viewing rate of the functional system exceeded the viewing rate of the sticky note system in any case. This difference in the viewing rate can be said to be the oversight rate of annotations when using the sticky note type system. By using this functional system, it is possible to reduce oversight of annotations given by members of the group and users themselves, and to utilize accumulated annotations more effectively.

5 IMPROVING THE SYSTEM

5.1 Comparison of sticky note type and functional type

The annotation sharing system described in the previous section is a functional type that registers annotations to words on an arbitrary page. Table 7 shows the results of comparing the annotation characteristics of the functional type and the conventional sticky note type annotation sharing system.

Focusing on the disadvantage of functional types, that is, the display of unhelpful postings, we conducted a preliminary experiment to see from what perspective the usefulness was judged.

5.2 Preliminary experiments

In the preliminary experiment, ten members of our laboratory actually submitted their annotations over a period of two weeks. In order to facilitate the comparison of annotations for the same word, they focused on two specified news articles as the target web pages for the experiment.

From this experiment, we found the following.

- Annotations with abstract or emotional content do not make sense when viewed on other pages.
- Users want to know on which page it was originally annotated.

- Many posts contain URLs to other pages.

Here, "abstract or emotional content" refers to annotations that do not refer to the target, such as difficult or interesting. In this experiment, we found three possible problems in the functional annotation sharing system. First, when the URL of another page is included in the annotation, is the page reliable? Second, annotations that have been around for a long time are not as useful as those that are more recent. Finally, if the annotation is not related to the word to which it is attached, its usefulness in browsing other pages is reduced.

5.3 Usability judgment method

Based on the problems identified in the previous section, we evaluate the annotation submissions from multiple viewpoints and judge their usefulness. Then, we propose a method of displaying only the top few annotations ranked by the judgment results when they are shared.

The following three criteria are used to judge the usefulness of this method.

- Time between submission and viewing.
- Relevance of annotations, added words, and the content of the page being viewed.
- Credibility of the page if the post contains URLs of other pages.

As the second criterion, we propose a method to determine the relevance of annotations and added words to the contents of the page being viewed. We first considered the relevance of annotations and added words, then annotations and browsed pages, but found that words and annotations are short and the accuracy of determining relevance is low. Therefore, we decided to use the relevance of the content of the page at the time of annotation registration and the page at the time of browsing. To determine the relevance, we measure the proximity of the topics of each content. To obtain these topics as numerical values, we use LDA [5], which is a topic model method. The number of topics at this time is not fixed, and we plan to evaluate the difference in results depending on the number of topics. In this work, we used LDA, which is an approach based on the frequency of occurrence of words, but we will also implement methods using Word2vec, which is an approach based on the distributed representation of words, and Doc2vec, which is an approach based on the distributed representation of sentences.

In the case of annotations with abstract or emotional contents mentioned in the problems, we are also considering a method to add a summary of the web page where the annotation was posted at the time of sharing. In this case, we plan to extract words expressing emotions from the annotation contents using natural language processing.

5.4 Embedding judgement method into the system

The first step is to calculate the topic of the web page that was used when the annotation was submitted. The values of the topics are stored in the database along with the contents of the annotation. When sharing the annotations, we calculate the closeness between

the topic of the web page being viewed and the topic of each word appearing in the web page.

6 RELATED WORK

6.1 Annotation sharing system

PAMS2.0 [13] developed by Su et al. is an annotation sharing system similar to this work. This system was developed mainly for education and aims to deepen understanding by writing and sharing annotations while reading the same teaching material within a limited group of students. In PAMS2.0, it is possible to chat individually or discuss with the whole group. It is also possible to attach annotations to tags such as questions and answers, and write text directly on the page.

Similarly, a system for educational purposes is MyNote [3] developed by Yu-Chien Chen et al. This system is built into the e-learning management system (LMS). It can annotate learning objects in the LMS, and also annotate Web documents from the LMS.

In contrast to the above research, which is a sticky note type that adds an annotation to a specific page, the system of this research is a function type that registers an annotation in the word itself on an arbitrary page.

6.2 Usability judgement method

In terms of evaluating the contents posted by an unspecified number of users, the studies related to this research include the detection of fake news in Twitter posts and the evaluation of reviews on e-commerce sites.

Siva Charan Reddy Gangireddy et al.'s research [6] uses graph-based unsupervised learning by assuming the behavior of users who send fake news. Qiang Zhang et al. [14] proposed a Bayesian deep learning model for fake news detection by considering the content and time series of posts and replies. Debanjan Paul et al. [12] proposed a review evaluation method using a dynamic convolutional neural network, focusing on the fact that the conventional review evaluation method for e-commerce sites uses other users' ratings and cannot correctly evaluate reviews with few ratings from other users. They also used natural language processing to deal with the fact that different words are used when reviewing the same aspect.

7 CONCLUSION AND FUTURE WORK

In this paper, we implemented a functional annotation sharing system to facilitate sharing of information and recognition within a limited group. By annotating the words themselves rather than specific pages, the viewing rate for using annotations more effectively was increased.

However, since functional types may display unhelpful posts, we are developing a usefulness judgment method to solve this problem.

Currently, we are only implementing the second criterion, which is to judge the relevance of annotations, added words, and the content of the page being viewed. Therefore, in order to improve the accuracy of the usefulness judgment, it is our future task to implement the other three criteria in the system. In addition, we will also consider how to weight and rank the data calculated by each criterion. For the first point, judging the usefulness based on the period between posting and browsing, we plan to implement

a system that obtains the date and time of posting and the date and time of browsing, and calculates the difference. As for the third point, the reliability of other pages, we will consider how to determine this in the future.

REFERENCES

- [1] Adobe. 2021. *Adobe Acrobat DC*. Retrieved July 1, 2021 from <https://acrobat.adobe.com/jp/ja/acrobat.html>
- [2] A.nnotate. 2008. *Upload, Annotate, Share. Online document review and collaboration - PDF, Word and HTML*. Retrieved July 5, 2021 from <http://a.nnotate.com>
- [3] Yu-Chien Chen, Ren-Hung Hwang, and Cheng-Yu Wang. 2012. Development and evaluation of a Web 2.0 annotation system as a learning tool in an e-learning environment. In *Computers & Education* 58. 1094–1105.
- [4] Diigo. 2006. *Better reading and research with annotation, highlighter, sticky notes, archiving, bookmarking & more*. Retrieved July 5, 2021 from <https://www.diigo.com/>
- [5] D.M.Blei, A.Y.Ng, and M.I.Jordan. 2003. Latent Dirichlet Allocation. In *Machine Learning Research*, Vol. 3. 993–1022.
- [6] Siva Charan Reddy Gangireddy, Deepak P, Cheng Long Nanyan, and Tanmoy Chakraborty. 2020. Unsupervised Fake News Detection: A Graph-based Approach. In *Proceedings of the 31st ACM Conference on Hypertext and Social Media*. 75–83.
- [7] Masahiro Hayashi, Shun Aoyama, Sungmin Joo, and Motomichi Toyama. 2011. Keio WIX system(1) User Interface. In *DEIM2011 The 3rd Forum on Data Engineering and Information Management*. Shizuoka, Japan.
- [8] Fuminori Ishizaki and Motomichi Toyama. 2012. An Incremental Update Algorithm for Large Aho-Corasick Automaton. In *DEIM2012 The 4th Forum on Data Engineering and Information Management*. Kobe, Japan.
- [9] Ryosuke Mori, Tatsuya Yabu, Sungmin Joo, and Motomichi Toyama. 2011. Keio WIX system(2) Server-side implementation. In *DEIM2011 The 3rd Forum on Data Engineering and Information Management*. Shizuoka, Japan.
- [10] BBC NEWS. 2020. *Coronavirus: Google Duplex AI arrives in UK to ask about opening hours*. Retrieved July 2, 2021 from <https://www.bbc.com/news/technology-52232121>
- [11] BBC NEWS. 2020. *How important will blockchain be to the world's economy?* Retrieved July 2, 2021 from <https://www.bbc.com/news/business-48526666>
- [12] Debanjan Paul, Sudeshna Sarkar, and Muthusamy Chelliah. 2017. Recommendation of High Quality Representative Reviews in e-commerce. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 311–315.
- [13] Addison Y.S. Su, Stephen J.H. Yang, Wu-Yuin Hwang, and Jia Zhang. 2010. A Web 2.0-based collaborative annotation system for enhancing knowledge sharing in collaborative learning environments. In *Computers & Education* 55. 752–756.
- [14] Qiang Zhang, Aldo Lipani, Shangsong Liang, and Emine Yilmaz. 2019. Reply-Aided Detection of Misinformation via Bayesian Deep Learning. In *Proceedings of the 2019 World Wide Web Conference (WWW '19)*. ACM, 2333–2343.

Sentimental Analysis Applications and Approaches during COVID-19: A Survey

Areeba Umair
Department of Electrical Engineering
and Information Technologies,
University of Naples Federico II
Naples, Italy
areeba.umair@unina.it

Elio Masciari
University of Naples Federico II,
ICAR-CNR – Institute of Italian
National Research Council
Italy
elio.masciari@unina.it

Muhammad Habib Ullah
Department of Electrical Engineering
and Information Technologies,
University of Naples Federico II
Naples, Italy
muhammad.habibullah@unina.it

ABSTRACT

The social media and electronic media has a vast amount of user-generated data such as people's comment and reviews about different product, diseases, government policies etc. Sentimental analysis is the emerging field in text mining where people's feeling and emotions are extracted using different techniques. COVID-19 has declared as pandemic and effected people's lives all over the globe. It caused the feelings of fear, anxiety, anger, depression and many other psychological issues. In this survey paper, the sentimental analysis applications and methods which are used for COVID-19 research are briefly presented. The comparison of thirty primary studies shows that Naive Bayes and SVM are the widely used algorithms of sentimental analysis for COVID-19 research. The applications of sentimental analysis during COVID includes the analysis of people's sentiments specially students, reopening sentiments, analysis of restaurants reviews and analysis of vaccine sentiments.

CCS CONCEPTS

• **Computing methodologies** → **Classification and regression trees**; • **Applied computing** → **Multi-criterion optimization and decision-making**; • **Information systems** → **Sentiment analysis**; • **Human-centered computing** → **Heat maps**; • **Social and professional topics** → **Employment issues**.

KEYWORDS

Big Data, Sentimental Analysis, Social Media, Machine Learning, COVID-19, Coronavirus, People's reviews

ACM Reference Format:

Areeba Umair, Elio Masciari, and Muhammad Habib Ullah. 2021. Sentimental Analysis Applications and Approaches during COVID-19: A Survey. In *25th International Database Engineering Applications Symposium (IDEAS 2021)*, July 14–16, 2021, Montreal, QC, Canada. ACM, New York, NY, USA, 5 pages. <https://doi.org/10.1145/3472163.3472274>

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

IDEAS 2021, July 14–16, 2021, Montreal, QC, Canada

© 2021 Association for Computing Machinery.

ACM ISBN 978-1-4503-8991-4/21/07...\$15.00

<https://doi.org/10.1145/3472163.3472274>

1 INTRODUCTION

Today, people usually make review on internet to express their opinion about the service, things they use or about any trending topic, [34]. With the advancement of technologies and industrialization era, a number of industries have been setting up and appealing the customers to buy their products online and also make review to highlight their positive and negative experience about their products. The overwhelming of review data on internet has increased the chances of extracting valuable information from it in the form of sentiments [15] [16]. Sometimes, people express multiple sentiments polarities in a single text. It is good for a business or organization to understand the sentiments of the customers. Sentimental analysis is natural language processing (NLP) task which usually understands the meaning of the text [14] and classify in positive, negative or neutral sentiments of different aspects in a text. The aspect level sentimental analysis is the fine-grain task in sentimental analysis and provides a complete sentimental expression [34]. For instance, "The chair is comfortable but its price is high". In this example, "comfortable" and "high" represent positive and negative sentiments respectively. The advancement in artificial intelligence techniques has eliminates the traditional methods of sentimental analysis.

After World War II, COVID-19 has considered as the biggest problem faced by whole world. The coronavirus was first spotted in China in December 2019 and has spread in the whole world. According to John Hopkins University, more than 130 million people have been affected and 2,861,677 number of deaths have been occurred due to COVID-19 till the first week of April 2021. The destabilization caused by COVID has created the multiple emotions in people such as fear, anger, anxiety, depression even hostility [31].

The two ways to solve sentimental classification tasks are traditional machine learning methods and deep learning methods. The traditional methods usually use classifiers i.e. SVM (Support Vector Machine) and Naive Bayes for this purpose while in deep learning methods Recurrent Neural Network (RNN) and Convolutional Neural Network (CNN) have widely used in NLP tasks. Deep learning is becoming famous in this regard as it perform feature engineering and extract meaning features automatically [34]. The literature shows that CNN does not perform well in capturing the sequential dependencies and RNN suffers from gradient vanishing and parallel-unfriendly problems due to its recurrent nature. This indicates that the existing approaches for sentimental classifications have shorts comings such as low performance in terms of accuracy and recall and high training timings etc. When there are multiple aspects with inconsistent sentiment polarity in a sentence,

the dependencies between the words will be weakened due to increase in distance. In such case, the attention mechanism can put its focus on the most important information.

In this study, we have performed the survey of thirty primary studies related to sentimental analysis during COVID-19 pandemic and figure out the techniques that have been applied in order to classify the sentiments of the people as well as the application areas of sentimental analysis during COVID research. The objectives of this survey are to identify the data sources and data volume of sentimental analysis during COVID-19, to identify the mostly used approaches and the applications of sentimental analysis during COVID. This study also presents the future implications of research with respect to COVID.

2 METHODOLOGY:

The review of thirty primary studies has been conducted in this study as shown in Table 1. In Table 1, benchmark data sets and well known data sources are mentioned in column 2 to help the researchers or readers in getting similar kind of data. The volume of data used in individual study has been mentioned in column 3. The Column 4 specifies the types of approaches or techniques which have been widely used during COVID for sentimental analysis and classification. During COVID, sentimental analysis was performed over different application areas which have been illustrated in column 5 of Table 1. This is the most important aspect of this surveys as it can open new research directions or topics for future researchers. The future trends and implications have been presented in column 6.

2.1 Data Sources during COVID-19 research:

Sentimental analysis is considered as a sentimental classification task. During COVID-19 pandemic, people experience different emotions and express their emotions using different social media platform. The social media platforms are the rich source of information as well as data in order to figure out the people's reactions and feelings during the destruction of COVID. Table 1 shows that the biggest data source for the research during pandemic was **twitter**. The statistics shows that 24 out of 30 studies uses twitter as a data source while other sources of data are online media and forums, Weibo account, WeChat account, Reddit, Yelp, RateMDs, HealthGrades, and Vitals and Qingbo Big Data Agency. The information which can be found on these popular social media is given in table ??.

Twitter: Twitter is considered as most popular social media platform having almost 81.47 million registered users [1]. People share message, that are called "tweets", related to public and global situations ultimately turning the twitter into data hotspot for web-based media conversation. In a single day, people post about 500 million tweets which results in 200 billion tweets posted per year [4]. The tweets are grouped based on their topics such as political matters, personal opinion, national economic issues, COVID-19 pandemic [9].

WeChat: WeChat is the Chinese multi-purpose social media and messaging platform. It has one billion monthly active users which regarded the WeChat as most popular social media platform.

2.2 Approaches for COVID Sentimental Classification

With the rise of big data, there is a need to develop efficient analytics tools [10]. Sentimental Classification Approaches, during COVID-19 research, can be divided into three types. Machine learning based approaches, lexicon based approaches and hybrid approaches.

2.2.1 Machine Learning Approaches: The machine learning based approaches use the famous ML algorithms for the SC during COVID. They further consists of two categories i.e. supervised and unsupervised learning methods.

Supervised Learning Methods: In the supervised learning methods, the instances of the data are labelled already [30]. Various supervised learning methods have been used in literature for the sentimental classification in COVID-19 related research as seen in Table 1.

Naive Bayes: Naive Bayes is one of the supervised learning algorithm and have been used in [1] and [27]. It works on the principle of Bayesian theorem given in equation 1.

$$P(H|X) = P(X|H)P(H)/P(X) \quad (1)$$

Support Vector Machine: SVM is the statistical learning based machine learning algorithm that works by converting feature space into high dimensional features in order to find the hyperplane. It is used by [1], [20], [25] and [32].

Decision Tree and Random Forest: Decision tree is the machine learning algorithms that trains its model to predict the class values based on simple decision rules found in entire train dataset. Random Forest belongs to the family of decision tree and works by choosing random features as well as random instances. It has been used by [1], [12], [25] and [32].

Other supervised learning approaches used in SC for COVID research are KNN [1], Linear Regression [1], Logistic Regression [27], [32], LSTM [11], [25], RNN [19] and BERT model [4], [18] etc.

Unsupervised Learning Methods: In unsupervised learning methods, the data is not labelled. The unsupervised methods have been used in SC related to COVID. K-means clustering has been used by [6] while Latent Dirichlet Allocation (LDA) method has been used in many studies i.e. [5], [9], [22], [29], [32], [33], [35].

2.2.2 Lexicon Based Approaches: Two types of opinion words are used to express the feelings i.e. positive opinion words and negative opinion words which are used to express likes and dislikes respectively. Different approaches are used to collect the opinion words list.

Dictionary-based approach and Corpus-based approach Dictionary based and corpus based approaches have been used in [3] in order to perform the sentimental analysis during COVID-19

Natural Language Processing Natural language processing (NLP) is used along with lexicon based methods in order to find the semantic relationship in a sentence. It has been used in [2] for the mental health analysis of students during COVID. In [19] and [23], NLP has been used to Analyze sentiments and characteristics of Covid-19 respectively. [33] has also used NLP to examine COVID-19-related discussions, concerns, and sentiments using tweets.

2.3 Sentimental Analysis Applications during COVID-19

COVID-19 has attracted the researchers in the area of sentimental classification as COVID has effected people's behaviours and attitudes in many ways. There are various topics that work under sentimental classification during COVID-19.

2.3.1 Sentimental analysis on palliatives distribution during COVID-19. It is responsibility of any government to maintain the sustainability of any country. During COVID-19, people needed help in order to lessen the economic as well as psychological stress. For this purpose, different governments releases the relief packages and certain other bonuses. In developing countries, monitoring the public funds transparency is a challenge. Therefore, it is necessary for the government to analyse the people's reaction and sentiments on the palliative distribution as it will indicate the reach of funds and its impact on people's circumstances during COVID-19. In [1], Adamu et al. performed sentimental classification on Nigerian Government COVID-19 Palliatives Distribution

2.3.2 Public sentiments and mental health analysis of students during the lockdown. COVID-19 has stopped the lives of people as it spreads with the human-to-human interaction. To stop the spread of coronavirus, it is necessary to impose such steps which tend to stop the movement and results in lesser human-to-human interaction. The one measure which was adopted by almost all the states of the world is "lockdown", resulting in closing airspace, closing educational institutions and workplaces, closing public transport etc. Hence, these implications have caused sadness, loneliness, anxiety, fear and many other psychological issues in peoples specially in students. Some of the students have stuck in their hostels, far away from their hometowns, few students are worried because of their exams and educational activities. Thus, the lockdown has effected people's lives and emerged many physiological issues like depressions. During these days, people are using social media in order to express their feelings and emotions. These social media posts such as tweets can be analyzed and helps the researchers to understand the state-of-mind of the citizens [5], [22], [3] and students [2]. In [6], [19], [24], [25], [27], [23], [13], [29], [33], [9], [28] sentimental analysis of people's behaviour and attitudes during COVID-19 has been performed using twitter data. In [32], the sentimental analysis of tweets is carried out with respect to the age of the social media users and they found out the extent of tweets is higher in youth during COVID.

2.3.3 COVID-19 reopening sentiments: With the paradigm shift due to COVID-19, billions of people's life has been effected directly or indirectly. COVID-19 has induced the feelings of fear, anxiety as well as economical crisis, which altogether are the challenges towards the reopening after COVID [26]. Long-term lock-down is not a solution, instead a threat for the economy of any country. COVID-19 has effected the life of the students as well as the working people. Considering this situation, everyone is craving for going back to normal life and physical activities [17]. Hence, in [17] and [26], the researchers tried to analyse the sentiments of the people towards reopening after COVID-19 disasters.

2.3.4 Analyzing online restaurant reviews. This era of e-commerce has enabled the customers to led a satisfy and quality life. The on-line reviews has helped the customers in decision-making. Online reviews are important for the restaurants as well because they are aligned with the star rating and one-star increase can earn a good revenue for the restaurant. During COVID-19, many restaurants got negative reviews for being the cause of COVID-spread, for not proper heating the outdoor area or for slow service. Therefore, it is important to analyse the customers' sentiments in order to improve their quality. The researchers analyzed the customers' sentiments which in-turn helped the customers as well as restaurants management to get good quality food and environment and maintain high quality respectively [12].

2.3.5 Vaccine sentiments and racial sentiments. In [18], researchers used concept drift in order to classify the sentiments of people associated with COVID vaccine. During COVID, a rise in prejudice and discrimination behaviour against Asian citizens have been seen. The researchers tries to describe variations in people attitudes towards racism before and after COVID-19 [20].

2.4 Future implications of research about COVID

Various future implications have been given below:

- Consider huge amount of data and explore multiple data sources and platforms.
- Analyze sentiments expressed in multiple languages.
- Real-time monitoring and visualizations should be performed.
- Consider socioeconomic factors and household information while analysing the sentiments of the people.
- Use deep learning approaches in future.
- Sentiments on different age-groups should be performed.
- Explore public trust and confidence in existing measures and policies, which are essential for their well-being.
- More precise location information can improve spatial analysis.
- More specific topics can be analyzed to help policy maker, government and local Communities during any emergency conditions.

3 COMPARISON OF STUDIES

In this study, the comparison of thirty primary studies have been performed and represented in the tabular form in Table 1.

Table 1: Sentimental Analysis Approaches and Applications During COVID-19 Pandemic

Ref	Data Source/set	Volume of Data	Techniques	Application	Limitation/ Future Direction
[1]	Twitter	9803 Tweets	KNN, RF, NB, SVM, DT, LR	Sentimental analysis on COVID-19 Palliatives Distribution	Large data, consider multiple language, real time classification
[2]	Twitter	330,841 tweets	NLP, bar graph,	Mental Health Analysis of Students	N/A
[4]	twitter	3090 tweets	BERT Model	Classifying the fake tweets	N/A
[5]	Twitter	410,643 tweets	Scatter plot, line chart, LDA	Public sentiments during the lockdown	Focus on English language, understand the perceptions and contexts related to negative sentiment
[3]	Twitter	6,468,526 tweets.	Dictionary-based methodology and corpus-based methodology	Sentiment Analysis During COVID-19	N/A
[6]	Online survey	N/A	clustering algorithm (k-means)	Understand adults' thoughts and behaviors	N/A
[8]	Online media and forums, Weibo account, WeChat account	N/A	Correlation analysis	Construct a framework of COVID-19 from five Dimensions i.e. epidemic, medical, governmental, public, and media responses	N/A
[9]	Twitter	N=1,001,380	Latent Dirichlet Allocation	Sentimental Analysis, Identify dominant topics during COVID	Population is not represented, real-time posting
[11]	reddit	563,079 Comments	LSTM	Uncover issues related to COVID-19 from public opinions	Evaluate other social media using hybrid fuzzy deep-learning techniques
[12]	Yelp	112,412 reviews	GBDT, RF, LSTM, SWEM	Analyzing online restaurant reviews	Different review platforms and restaurant locations.
[13]	Twitter	20,325,929 tweets	CrystalFeel	Examine worldwide trends of fear, anger, sadness, and joy	Expanding the scope to include other media platforms.
[7]	Twitter	500,000 tweets	TextBlob	Determining polarity and subjectivity in COVID tweets.	Explore other social media
[18]	Twitter	57.5M English	BERT	Concept drift on vaccine sentiments	Concept drift in real-time social media monitoring project
[19]	Twitter	N/A	NLP, RNN	Analyze sentiments and manifestations	Visualization, clustering and classification
[20]	Twitter	3,377,295	SVM	Changes in racial sentiment	Examine longer-term temporal changes in racial attitudes
[21]	Twitter	840,000 tweets	TextBlob, LDA,	Attitude of Indian citizens while discussing the anxiety, stress, and trauma	Analyzing how perception changes for different biographies
[23]	Twitter	57 454 tweets	NLP and text analysis	Analyse the characteristics of polish Covid-19	N/A
[24]	Twitter	370 tweets	subjectivity vs. polarity, WordCloud	Sentimental analysis for COVID	N/A

4 CONCLUSION

This survey paper reviewed the thirty primary studies related to COVID-19 sentimental analysis and presented comparison with respect to sentimental analysis data sources, techniques and applications. This article contributes to the sentimental analysis field by considering its applications in real-world scenarios. The article concludes that the sentimental analysis during COVID-19 is still an open fields and contains many interesting topics using advanced methods of machine learning and deep learning. This article reported that Naive Bayes and SVM are the widely used algorithm for sentimental analysis during COVID.

REFERENCES

- [1] Hassan Adamu, Syaheerah Lebai Lutfi, Nurul Hashimah Ahamed Hassain Malim, Rohail Hassan, Assunta Di Vaio, and Ahmad Sufril Azlan Mohamed. 2021. Framing twitter public sentiment on Nigerian government COVID-19 palliatives distribution using machine learning. *Sustain.* 13, 6 (2021). <https://doi.org/10.3390/su13063497>
- [2] Ashi Agarwal, Basant Agarwal, Priyanka Harjule, and Ajay Agarwal. 2021. *Mental Health Analysis of Students in Major Cities of India During COVID-19*. Springer Singapore. 51–67 pages. https://doi.org/10.1007/978-981-33-4236-1_4
- [3] V. Ajantha Devi and Anand Nayyar. 2021. *Evaluation of Geotagging Twitter Data Using Sentiment Analysis During COVID-19*. Vol. 166. Springer Singapore. 601–608 pages. https://doi.org/10.1007/978-981-15-9689-6_65
- [4] Nalini Chintalapudi, Gopi Battineni, and Francesco Amenta. 2021. Sentimental Analysis of COVID-19 Tweets Using Deep Learning Models. *Infect. Dis. Rep.* 13, 2 (2021), 329–339. <https://doi.org/10.3390/idr13020032>
- [5] Subasish Das and Anandi Dutta. 2020. Characterizing public emotions and sentiments in COVID-19 environment: A case study of India. *J. Hum. Behav. Soc. Environ.* 31, 1–4 (2020), 1–14. <https://doi.org/10.1080/10911359.2020.1781015>
- [6] S. W. Flint, A. Piotrkowicz, and K. Watts. 2021. Use of Artificial Intelligence to understand adults' thoughts and behaviours relating to COVID-19. *Perspect. Public Health* XX, X (2021), 1–8. <https://doi.org/10.1177/1757913920979332>
- [7] Kamaran H. Manguri, Rebaz N. Ramadhan, and Pshko R. Mohammed Amin. 2020. Twitter Sentiment Analysis on Worldwide COVID-19 Outbreaks. *Kurdistan J. Appl. Res.* (2020), 54–65. <https://doi.org/10.24017/covid.8>
- [8] Hao Huang, Zongchao Peng, Hongtao Wu, and Qihui Xie. 2020. A big data analysis on the five dimensions of emergency management information in the early stage of COVID-19 in China. *J. Chinese Gov.* 5, 2 (2020), 213–233. <https://doi.org/10.1080/23812346.2020.1744923>
- [9] Man Hung, Evelyn Lauren, Eric S. Hon, Wendy C. Birmingham, Julie Xu, Sharon Su, Shirley D. Hon, Jungweon Park, Peter Dang, and Martin S. Lipsky. 2020. Social network analysis of COVID-19 sentiments: Application of artificial intelligence. *J. Med. Internet Res.* 22, 8 (2020), 1–13. <https://doi.org/10.2196/22590>
- [10] Michele Ianni, Elio Masciari, Giuseppe M. Mazzeo, Mario Mezzanzanica, and Carlo Zaniolo. 2020. Fast and effective Big Data exploration by clustering. *Future Gener. Comput. Syst.* 102 (2020), 84–94. <https://doi.org/10.1016/j.future.2019.07.077>
- [11] Hamed Jelodar, Yongli Wang, Rita Orji, and Hucheng Huang. 2020. Deep sentiment classification and topic discovery on novel coronavirus or COVID-19 online discussions: NLP using LSTM recurrent neural network approach. *arXiv* 24, 10 (2020), 2733–2742.
- [12] Yi Luo and Xiaowei Xu. 2021. Comparative study of deep learning models for analyzing online restaurant reviews in the era of the COVID-19 pandemic. *Int. J. Hosp. Manag.* 94, December 2020 (2021), 102849. <https://doi.org/10.1016/j.ijhbm.2020.102849>
- [13] May Oo Lwin, Jiahui Lu, Anita Sheldenkar, Peter Johannes Schulz, Wonsun Shin, Raj Gupta, and Yiping Yang. 2020. Global sentiments surrounding the COVID-19 pandemic on Twitter: Analysis of Twitter trends. *JMIR Public Heal. Surveill.* 6, 2 (2020), 1–4. <https://doi.org/10.2196/19447>
- [14] Giuseppe Manco, Elio Masciari, and Andrea Tagarelli. 2002. A Framework for Adaptive Mail Classification. In *14th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2002)*, 4–6 November 2002, Washington, DC, USA. IEEE Computer Society, 387. <https://doi.org/10.1109/TAL.2002.1180829>
- [15] Elio Masciari. 2009. Trajectory Clustering via Effective Partitioning. In *Flexible Query Answering Systems, 8th International Conference, FQAS 2009, Roskilde, Denmark, October 26–28, 2009. Proceedings (Lecture Notes in Computer Science, Vol. 5822)*, Troels Andreassen, Ronald R. Yager, Henrik Bulskov, Henning Christiansen, and Henrik Legind Larsen (Eds.). Springer, 358–370. https://doi.org/10.1007/978-3-642-04957-6_31
- [16] Elio Masciari. 2012. SMART: Stream Monitoring enterprise Activities by RFID Tags. *Inf. Sci.* 195 (2012), 25–44. <https://doi.org/10.1016/j.ins.2012.01.041>
- [17] Md Mokhesur Rahman, G. G.Md Nawaz Ali, Xue Jun Li, Kamal Chandra Paul, and Peter H.J. Chong. 2020. Twitter and Census Data Analytics to Explore Socioeconomic Factors for Post-COVID-19 Reopening Sentiment. *arXiv* (2020). <https://doi.org/10.2139/ssrn.3639551>
- [18] Martin Müller and Marcel Salathé. 2020. Addressing machine learning concept drift reveals declining vaccine sentiment during the COVID-19 pandemic. *arXiv* (2020), 1–12. [arXiv:2012.02197](https://arxiv.org/abs/2012.02197)
- [19] László Nemes and Attila Kiss. 2021. Social media sentiment analysis based on COVID-19. *J. Inf. Telecommun.* 5, 1 (2021), 1–15. <https://doi.org/10.1080/24751839.2020.1790793>
- [20] Thu T. Nguyen, Shaniece Criss, Pallavi Dwivedi, Dina Huang, Jessica Keralis, Erica Hsu, Lynn Phan, Leah H. Nguyen, Isha Yardi, M. Maria Glymour, Amami M. Allen, David H. Chae, Gilbert C. Gee, and Quynh C. Nguyen. 2020. Exploring U.S. shifts in anti-Asian sentiment with the emergence of COVID-19. *Int. J. Environ. Res. Public Health* 17, 19 (2020), 1–13. <https://doi.org/10.3390/ijerph17197032>
- [21] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Improving Language Understanding by. *OpenAI* (2018), 1–10. https://gluebenchmark.com/leaderboard/%0Ahttps://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf
- [22] S. V. Praveen, Rajesh Ittamalla, and Gerard Deepak. 2021. Analyzing Indian general public's perspective on anxiety, stress and trauma during Covid-19 - A machine learning study of 840,000 tweets. *Diabetes Metab. Syndr. Clin. Res. Rev.* 15, 3 (2021), 667–671. <https://doi.org/10.1016/j.dsx.2021.03.016>
- [23] Eryka Probiez, Adam Galuszka, and Tomasz Dzida. 2021. Twitter text data from #Covid-19: Analysis of changes in time using exploratory sentiment analysis. *J. Phys. Conf. Ser.* 1828, 1 (2021). <https://doi.org/10.1088/1742-6596/1828/1/012138>
- [24] Supriya Raheja and Anjani Asthana. 2021. Sentimental analysis of twitter comments on COVID-19. *Proc. Conflu. 2021 11th Int. Conf. Cloud Comput. Data Sci. Eng.* (2021), 704–708. <https://doi.org/10.1109/Confluence51648.2021.9377048>
- [25] Furqan Rustam, Madiha Khalid, Waqar Aslam, Vaibhav Rupapara, Arif Mehmood, and Gyu Sang Choi. 2021. A performance comparison of supervised machine learning models for Covid-19 tweets sentiment analysis. *PLoS One* 16, 2 (2021), 1–23. <https://doi.org/10.1371/journal.pone.0245909>
- [26] Jim Samuel, G. G.Md Nawaz Ali, Md Mokhesur Rahman, Ek Esawi, and Yana Samuel. 2020. COVID-19 public sentiment insights and machine learning for tweets classification. *Inf.* 11, 6 (2020), 1–22. <https://doi.org/10.3390/info11060314> [arXiv:2005.10898](https://arxiv.org/abs/2005.10898)
- [27] Jim Samuel, Md Mokhesur Rahman, G. G.Md Nawaz Ali, Yana Samuel, Alexander Pelaez, Peter Han Joo Chong, and Michael Yakubov. 2020. Feeling Positive about Reopening? New Normal Scenarios from COVID-19 US Reopen Sentiment Analytics. *IEEE Access* 8 (2020), 142173–142190. <https://doi.org/10.1109/ACCESS.2020.3013933>
- [28] Md Shahriare Satu, Md Imran Khan, Mufti Mahmud, Shahadat Uddin, Matthew A. Summers, Julian M.W. Quinn, and Mohammad Ali Moni. 2020. TClustVID: A novel machine learning classification model to investigate topics and sentiment in COVID-19 tweets. *medRxiv* (2020). <https://doi.org/10.1101/2020.08.04.20167973>
- [29] Adnan Muhammad Shah, Xiangbin Yan, Abdul Qayyum, Rizwan Ali Naqvi, and Syed Jamal Shah. 2021. Mining topic and sentiment dynamics in physician rating websites during the early wave of the COVID-19 pandemic: Machine learning approach. *Int. J. Med. Inform.* 149, February (2021). <https://doi.org/10.1016/j.ijmedinf.2021.104434>
- [30] Areeba Umair, Muhammad Shahzad Sarfraz, Muhammad Ahmad, Usman Habib, Muhammad Habib Ullah, and Manuel Mazzara. 2020. applied sciences Spatiotemporal Analysis of Web News Archives for Crime Prediction. (2020). <https://doi.org/10.3390/app10228220>
- [31] Nishant Vishwamitra, Ruijia Roger Hu, Feng Luo, Long Cheng, Matthew Costello, and Yin Yang. 2020. On Analyzing COVID-19-related Hate Speech Using BERT Attention. *Proc. - 19th IEEE Int. Conf. Mach. Learn. Appl. ICMLA 2020* (2020), 669–676. <https://doi.org/10.1109/ICMLA51294.2020.00111>
- [32] Xiaoling Xiang, Xuan Lu, Alex Halavanau, Jia Xue, Yihang Sun, Patrick Ho Lam Lai, and Zhenke Wu. 2021. Modern Senicide in the Face of a Pandemic: An Examination of Public Discourse and Sentiment About Older Adults and COVID-19 Using Machine Learning. *J. Gerontol. B. Psychol. Sci. Soc. Sci.* 76, 4 (2021), e190–e200. <https://doi.org/10.1093/geronb/gbaa128>
- [33] Jia Xue, Junxiang Chen, Chen Chen, Chengda Zheng, Sijia Li, and Tingshao Zhu. 2020. Public discourse and sentiment during the COVID 19 pandemic: Using latent dirichlet allocation for topic modeling on twitter. *PLoS One* 15, 9 September (2020), 1–12. <https://doi.org/10.1371/journal.pone.0239441> [arXiv:2005.08817](https://arxiv.org/abs/2005.08817)
- [34] Ziyu Zhou, Fang Ai Liu, and Qianqian Wang. 2019. R-Transformer network based on position and self-attention mechanism for aspect-level sentiment classification. *IEEE Access* 7 (2019), 127754–127764. <https://doi.org/10.1109/ACCESS.2019.2938854>
- [35] Bangren Zhu, Xinqi Zheng, Haiyan Liu, Jiayang Li, and Peipei Wang. 2020. Analysis of spatiotemporal characteristics of big data on social media sentiment with COVID-19 epidemic topics. *Chaos, Solitons and Fractals* 140 (2020), 110123. <https://doi.org/10.1016/j.chaos.2020.110123>

Author List

Afrati, Foto 1	Fragkou, Evangelia 21
Ait El Cadi, Abdessamad 263	Fuduli, Antonio 291
Aligon, Julien 194	Fuller, Clifton 273
And Olga Ormandjieva, Juan J. Cuadrado-gallego 284	Fung, Daryl 66
Avolio, Matteo 291	Garrido, Angel 76
Bellatreche, Ladjel 27	Ghedira Guegan, Chirine 279
Bergami, Giacomo 222	Goto, Kento 173
Bhardwaj, Dave 284	Groppe, Sven 149
Bhise, Minal 127	Groppe, Jinghua 149
Bills, Joseph 212	Gupta, Nidhi 127
Bobed, Carlos 76	Habib Ullah, Muhammad 303
Boukadi, Khouloud 279	Holubova, Irena 134, 242
Canahuate, Guadalupe 273	Kabachi, Nadia 279
Chevalier, Max 184	Katsaros, Dimitrios 21
Comito, Carmela 104	Kechar, Mohamed 27
Contos, Pavel 134, 242	Leung, Carson 66
Damigos, Matthew 1	Li, Jingrui 173
Dang, Vincent-nam 95	Lietz, Kristina 85
Darmont, Jérôme 232, 252	Liu, Chuan-Ming 204
Delot, Thierry 263	Liu, Pengfei 252
Desai, Bipin 112	Loudcher, Sabine 252
Desai, Bipin 36	Mai, Daniel 66
Dyreson, Curtis 57	Manolopoulos, Yannis 21
Fatone, Gerardo 141	Mao, Yuhao 11
Ferrettini, Gabriel 194	Marai, Georgeta-Elisabeta 273
Flann, Nicholas 57	Marty, Joan 184

Author List(Continued)

Masciari, Elio 141, 303	Svoboda, Martin 134, 242
Megdiche, Imen 95, 194	Tamburis, Oscar 141
Mena, Eduardo 76	Teste, Olivier 232
Mohamed, Abdallah 273	Toyama, Motomichi 173, 296
Mokhov, Serguei 11	Tran, Jason 66
Moussa, Rim 157	Trépanier, Martin 263
Mudur, Sudhir 11	Umair, Areeba 303
Mufida, Miratul 263	Van Dijk, Lisanne 273
Nait-bahloul, Safia 27	Vidé, Bastien 184
Ng, Yiu-kai 212	Vocaturro, Eugenio 291
Noûs, Camille 252	Wang, Yaohua 273
Ortiz Castellanos, Ari 204	Wen, Qi 66
Pak, Denis 204	Wiese, Ingmar 85
Pandat, Ami 127	Wiese, Lena 85
Peiro, Alvaro 76	Yang, Yuzhao 232
Ravat, Franck 184	Zhang, Xinhua 273
Ravat, Franck 95, 194, 232	Zhao, Yan 95, 194
Revesz, Peter 47	Zouari, Firas 279
Roman, Cristian 76	Zumpano, Ester 291
Saeki, Yui 296	
Sahri, Soror 157	
Sharma, Vishal 57	
Soule-dupuy, Chantal 194	
Souza, Joglas 66	
Stasinopoulos, Nikos 1	
Suzuki, Yu 166	



UNIVERSITÉ
Concordia
UNIVERSITY



Published by ACM

ConfSys.org

ISBN 978-1-988392-07-3



9 781988 392073



10101 >